Radio elettronica & Computer

13 programmi per C64 e C128

Anno XVIII - N. 4 - Maggio 1989 - L. 8.500

Sped. in Abb. Postale Gr. III/70%

CALCOLO INTEGRALE
IL VIDEO-ESERCIZIARIO

PER CHI COMINCIA
LOGO TUTTO IN ITALIANO

UTILITY
UN PACKER PER 250 BLOCCHI

SIMULAZIONE
DUE AEREI, DUE JOYSTICK

SUONI
DIGITALIZZARLI CON AMIGA

Trasferimento automatico dei programmi da cassetta a disco

FAI DA TE
contro lo stress il rilassatore elettronico

Tassa pagata per campione allegato

Gruppo Editoriale JCE

RED HEAT
- DANKO -
L'Est e l'Ovest mettono insieme le loro forze per combattere un mercante di droga sovietico. Due detective - uno russo e uno di Chicago- sono spinti da motivi diversi, ma la preda é una sola.
- DANKO -
Il più caldo "Tie - in " del momento, tutto azione e grafica superba.

Radio Elettronica & Computer

N. 4 Maggio 1989

Sommario

Caricate così i programmi della cassetta allegata:

*Riavvolgete il nastro e digitate **LOAD** seguito da **RETURN** sulla tastiera del C64 e **PLAY** sul registratore. Verrà caricato il programma di presentazione con il menù dei programmi. Digitate **RUN** seguito dalla pressione del tasto **RETURN**. Terminata la presentazione, per caricare uno qualsiasi dei programmi è sufficiente digitare:*
LOAD "NOME PROGRAMMA"
*seguito dalla pressione del tasto **RETURN**.*

Responsabile Editoriale
Divisione Informatica
Francesca Marzotto

Direttore responsabile
Paolo Romani

Caporedattore
Fernando Zanini

Responsabile grafico
Desktop Publishing
Adelio Barcella

Impaginazione elettronica
Denise De Matteis

Segretaria di redazione
Alessandra Marini

Collaboratori
Paolo Gussoni, Dolma Poli, Isa Sestini

Foto di copertina: Maurizio Lodi/Ubik

RadioELETTRONICA&COMPUTER
Rivista mensile, una copia L. 8.500, numeri arretrati lire 10.000 cadauno.
Pubblicazione mensile registrata presso il Tribunale di Monza n. 679 del 28/11/88.

Fotolito: Bassoli - Milano.
Stampa: GEMM Grafica srl, Paderno Dugnano (MI).
Diffusione: Concessionario esclusivo per l'Italia A.&G. Marco SpA, Via Fortezza 27 - 20126 Milano.
Spedizione in abb. post. gruppo III/70.

Abbonamenti: Annuale L. 54.000, estero L. 83.000.
Biennale L. 97.500

RadioELETTRONICA & COMPUTER è titolare in esclusiva per l'Italia dei testi e dei progetti di Radio Plans e Electronique Pratique, periodici del gruppo Societé Parisienne d'Edition.

Gruppo Editoriale JCE srl
Sede legale, Direzione, Redazione, Amministrazione
Via Ferri 6 - 20092 Cinisello Balsamo (MI)
Tel. 02/61.73.441 - 61.72.671 - 61.72.641- 61.80.228
Telex 352376 JCE MIL I - Telefax 61.27.620

Direzione Amministrativa: Walter Buzzavo

Pubblicità e Marketing
Gruppo Editoriale JCE - Divisione Pubblicità
Via Ferri 6 - 20092 Cinisello Balsamo (MI)
Tel. 02/61.20.586 - 61.27.827 - 61.23.397 - 61.29.0038

Concessionario eslusivo per Roma, Lazio e centro sud:
UNION MEDIA srl - Via C. Fracassini, 18
00198 Roma - Tel. 06/3215434 (13 linee R.A.)
Telex 630206 UNION I - Telefax 06/3215678

Abbonamenti: Le richieste di informazioni sugli abbonamenti in corso si ricevono per telefono tutti i giorni lavorativi dalle ore 9 alle 12. Tel. 02/61.72.671 - 61.80.228 (int. 311-338).

Spedizioni: Daniela Radicchi

I versamenti vanno indirizzati a: Gruppo Editoriale JCE srl, Via Ferri 6 20092 Cinisello Balsamo (MI), mediante l'emissione di assegno circolare, cartolina vaglia o utilizzando il c.c.p. n. 351205. Per i cambi di indirizzo allegare alla comunicazione l'importo di L. 3.000, anche in francobolli, e indicare insieme al nuovo anche il vecchio indirizzo.

Associato al

Consorzio Stampa Specializzata Tecnica

Testata in corso di certificazione obbligatoria secondo quanto stabilito dal Regolamento del C.S.S.T.

Mensile associato all'USPI Unione Stampa Periodica Italiana

Audiomaster

Dedichiamo la parte iniziale di questa rubrica a un argomento nuovo: le novità software per Amiga. Poiché in questo stesso numero della rivista potete trovare la prova del digitalizzatore audio Easy Sound abbiamo pensato di completare l'argomento offrendovi la prova di un ottimo programma di digitalizzazione audio per Amiga, cioè di Audiomaster prodotto dalla Aegis Software, la stessa casa che ha prodotto l'ottimo programma musicale Sonix del quale potete trovare la prova sul numero di novembre di *Radio Elettronica & Computer.*

Audiomaster è un completo programma di digitalizzazione sonora che consiste in due processi diversi: la campionatura, che richiede l'intervento di un apposito accessorio hardware detto digitalizzatore audio, e l'editing del suono, che è un'operazione software.

Audiomaster: la rappresentazione grafica delle fluttuazioni di intensità e frequenza del suono campionato rende facilissima qualsiasi manipolazione.

Audiomaster è in grado di gestire in modo perfetto Easy Sound, il che significa che vi permette di trasferire qualunque brano musicale oppure qualunque rumore o effetto sonoro nella memoria RAM di Amiga. Dopo avere effettuato questa operazione il campionatore non serve più. Tutto il lavoro di montaggio del suono è affidato al computer e quindi ad Audiomaster.

Vediamo allora cos'è possibile effettuare tramite questo programma: innanzitutto la veste grafica, Audiomaster utilizza una finestra video di formato 640 x 200 pixel. Tale finestra è fissa, cioè non è consentito muoverla per lo schermo né modificarne le dimensioni, tuttavia esistono i gadget di priorità ed è quindi possibile portare altre finestre sopra quella di Audiomaster. Anche Audiomaster consente il multitasking, molto comodo per mantenere un CLI sempre disponibile nella parte inferiore dello schermo.

La parte centrale della finestra Audiomaster è riservata all'immagine della forma d'onda in memoria, che in seguito chiameremo riquadro onda. La larghezza del riquadro onda è variabile, infatti di default mostra l'intera onda campionata, ma sono disponibili alcuni comandi che consentono di visualizzarne solo una parte. Nella parte inferiore dello schermo si trovano i vari gadget che consentono di impartire alcuni comandi in modo molto veloce servendosi del mouse. La maggior parte dei comandi però si trova nei vari menù a discesa: molti comandi sono accessibili anche tramite la pressione del tasto Amiga. Ad esempio i comandi di editing classici Cut, Copy e Paste sono disponibili come di consueto con i tasti Amiga-X, Amiga-C e Amiga-P.

Con il comando Sampler compare la finestra di campionatura che consente di selezionare la frequenza di campionamento tra le due consentite (circa 8800 Hz e 17600 Hz) nonché l'ammontare di memoria destinato a mantenere i dati, che può variare da pochi kilobytes a oltre 300. Naturalmente se non si dispone di un'espansione di memoria è meglio non esagerare, altrimenti non si riesce a utilizzare le altre opzioni del programma.

Terminata la fase di campionatura si può passare all'editing, cioè alla manipolazione del suono. È possibile effettuare un editing classico, ovvero prendere pezzi di brano e comporli a proprio piacimento, ma le operazioni più spettacolari sono gli effetti speciali. Il più eclatante è l'effetto eco, per ottenere il quale è sufficiente selezionare la parte del brano campionato che deve essere elaborata, la velocità di ripetizione e il decremento dell'intensità. L'operazione è davvero banale, però per ottenere buoni effetti è necessaria un po' di pratica, poiché se si crea l'effetto eco a un brano troppo lungo o poco nitido non si ottiene altro che un suono confuso. Un altro effetto consiste nell'inversione temporale del suono: servendosi con una certa abilità di questo effetto è possibile ottenere ottimi scratching, meglio di un disc-jockey professionista.

Non manca il controllo della modulazione in ampiezza del suono, che permet-

te ottimi fading. Dulcis in fundo la funzione mix, che consente il mixing digitale di due suoni. Anche questa funzione è molto semplice da utilizzare in teoria, ma in realtà è necessario fare molta pratica per ottenere buoni risultati; inutile sottolineare che è possibile ottenere mixing davvero perfetti.

Tutte le funzioni viste finora trovano una vasta applicazione sia se si desidera campionare un brano lungo al massimo 30-40 secondi, sia se si desidera elaborare un suono singolo. Nel primo caso il lavoro è terminato, infatti basta salvare il brano campionato su floppy disk per poterlo utilizzare per qualunque applicazione. Naturalmente la registrazione è effettuata secondo lo standard IFF (Interchange File Format) che consente lo scambio del file con altri programmi musicali.

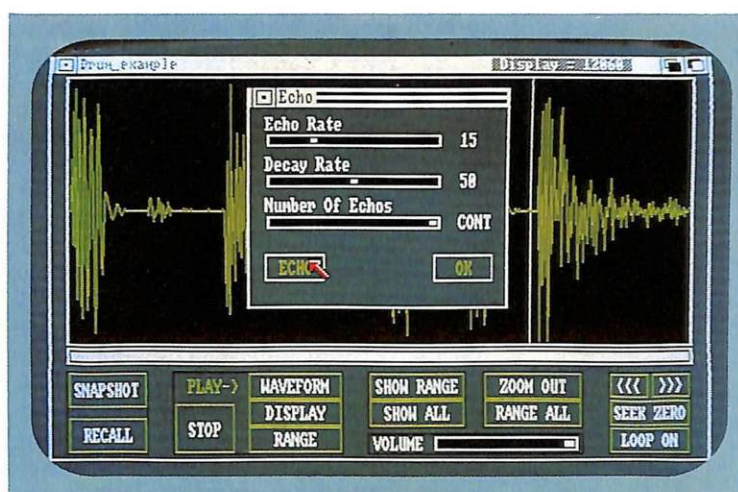
Una caratteristica davvero eccezionale di Audiomaster però è la possibilità di salvare il suono in formato compatibile con Sonix, il che significa che è possibile per esempio campionare il suono di un pianoforte (una nota sola) e poi suonarlo con Sonix, ovviamente esteso a ben cinque ottave. Sono disponibili molte opzioni per controllare questo tipo di elaborazione, in particolare può essere utile l'effetto loop che consente di utilizzare una nota che dura un solo istante estendendola al tempo desiderato.

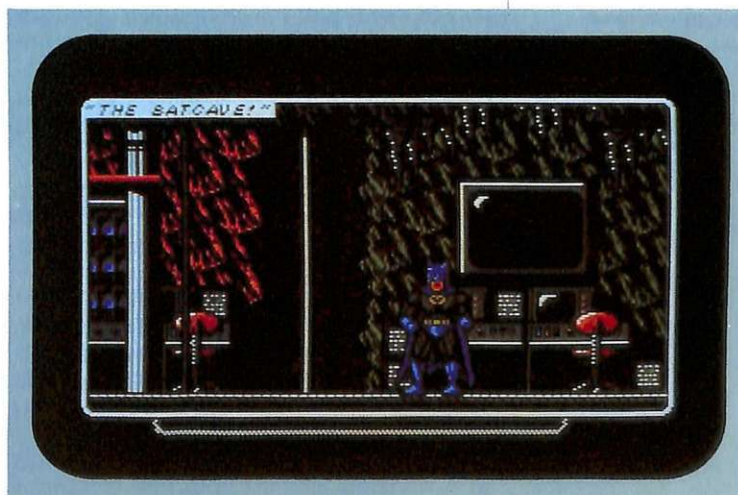
Ci sarebbe ancora molto da dire su Audiomaster in quanto si tratta di un programma davvero molto completo, efficace e facile da usare, vi consigliamo quindi di leggere la prova di Easy Sound presente in questo stesso numero della rivista, dove potrete trovare altre informazioni sulla campionatura dei suoni con Amiga.

Fra le tante manipolazioni del suono campionato esiste l'effetto eco per generare effetti sonori veramente speciali.

Giochi per Amiga

Anche il mercato dei videogiochi offre interessanti novità. Una particolare attenzione merita:

• **Batman, the Caped Crusader**, prodotto dalla Ocean e distribuito in Italia dalla Leader. Questo gioco è una via di mezzo tra un multiscreen e un adventure, infatti lo svolgimento del gioco è dinamico come per un vero multiscreen, tuttavia la ricchezza delle situazioni fa pensare a un sofisticato adventure. Il protagonista è il famosissimo Batman, l'eroe dei fumetti, e il luogo dell'azione non può essere altro che Gotham City.

Il gioco si presenta molto ricco, infatti è possibile scegliere tra due diverse avventure: "Il complotto di Penguin" e "Il complotto di Joker", ovvero la lotta ai due nemici storici di Batman. Nel primo caso Penguin è appena stato rilasciato dalla prigione e decide di aprire una fabbrica di ombrelli a Gotham City. Naturalmente questa attività serve solo da copertura per i suoi nefandi progetti che consistono nella conquista del mondo tramite un esercito di pinguini armati. Nel secondo caso il problema è la misteriosa sparizione di Robin, il compagno di avventure di Batman; al suo posto è rimasta solo una carta da gioco raffigurante un jolly, ovvero la firma del perfido Joker. La carta viene esaminata al Batcomputer che rivela l'inquietante messaggio: "Robin è diretto

Batman: una fase del gioco. Quando il supereroe cambia stanza o ambiente si apre una nuova finestra che si sovrappone a quelle precedenti.

verso un destino peggiore della morte".

Il gioco si svolge utilizzando il mouse, o meglio del joystick, che controlla tutti i movimenti di Batman, sia gli spostamenti, sia i colpi. È anche disponibile uno schermo funzioni dal quale è possibile ottenere informazioni sull'andamento dell'operazione, nonché utilizzare i vari oggetti che si incontrano durante l'avventura. Batman è un gioco appassionante, certamente buona parte del fascino del gioco è da attribuirsi al personaggio utilizzato: un anonimo esserino non avrebbe assolutamente fornito lo stesso effetto. La grafica è semplice, ma il movimento di Batman e degli altri personaggi è molto bello, a parte il fatto che il movimento degli sprites rispetto al fondo non è perfettamente sincronizzato con l'alternanza che ne simula il movimento, e quindi sembra sempre che i piedi scivolino sul pavimento. I personaggi comunque sono piuttosto grandi e sono disegnati con un certo dettaglio che conferisce loro un buon realismo. Il sonoro è di buona qualità, ma è un poco ossessivo. Non manca comunque un'icona che ne permette il disinserimento.

F14 Tomcat: la simulazione inizia con una specie di piccolo esame sui vari componenti dell'aereo.

Giochi per Commodore

Le novità software per il 64 che questo mese vi proponiamo sono quattro: due sono delle vere e proprie compilation di giochi sportivi mentre le altre sono un simulatore di volo e un classico arcade.

• **F-14 Tomcat** è un gioco d'azione della Activision ispirato alla più rinomata scuola di aviazione militare americana, la famosa Top Gun School. Durante il gioco percorrerete la lunga e difficile carriera che vi porterà a diventare, da semplice matricola, un infallibile asso del cielo. Farete i primi passi a bordo di un comune aereo da turismo poi, con spirito di abnegazione e tanto allenamento, raggiungerete la cabina di pilotaggio di un vero F-14 Tomcat, il caccia più maneggevole e micidiale oggi in circolazione.

Il pilota di un Tomcat ha a disposizione sistemi multipli di attacco e di difesa che consistono in tre diversi tipi di missili, ognuno con differente gittata, una mitragliatrice, un sistema di intercettazione radar e una sofisticatissima strumentazione per mantenere sotto controllo il volo. Tutto ciò è riproposto in ogni particolare in questa simulazione; la completezza degli strumenti di bordo gioca a favore del realismo e sicuramente non interviene a scapito della giocabilità.

Infatti sia la lettura delle informazioni necessarie al volo che l'accesso ai diversi sistemi d'offesa sono sempre semplici e immediati.

Una volta caricato il programma vi si presentano tre possibilità di gioco: sceglieté la prima se desiderate costruire da zero la vostra carriera di pilota di caccia, la seconda per continuare una carriera già intrapresa e registrata sul data disk, la terza se volete cimentarvi in un combattimento aereo in uno degli ottanta punti più caldi della terra.

Il corso di volo, necessario per poter iniziare la carriera, è composto da tre fasi: la prima, in cui imparerete manovre fondamentali come il cambio di quota o il dirigersi su una particolare rotta, vi permetterà, in caso di esito favorevole, di accedere al secondo livello di tirocinio, un combattimento simulato contro il vostro capitano. Superata anche questa prova sarete promossi di grado e arriverete al livello finale in cui affinerete la vostra abilità nei combattimenti assistiti da un altro caccia. Vi verrà assegnato un ufficiale di intercettazione radar che siederà nella postazione posteriore del vostro aviogetto e vi aggiornerà continuamente sulle rotte e sulle tattiche. A questo punto sarete a tutti gli effetti un pilota di F-14 e incomincerete il vostro servizio a bordo della portaerei Nimitz.

La natura di ogni missione dipende dal luogo in cui l'ammiraglia della flotta americana si trova, dalla situazione di

crisi in atto e dal numero e dalle intenzioni degli aerei ostili in avvicinamento. L'ammiraglio in persona vi darà istruzioni su come comportarvi in caso di attacco.

Gli esiti delle missioni possono essere quattro e ognuno influisce negativamente o positivamente sul punteggio, sul grado raggiunto e sulle onorificenze acquisite: potrete completare la missione, essere costretti a catapultarvi fuori dall'aviogetto e ad ammarare col paracadute, ritornare sani e salvi senza però aver esattamente adempiuto agli ordini o, nel peggiore dei casi, non ritornare affatto.

F-14 Tomcat riuscirà sicuramente a soddisfare sia le esigenze del patito delle simulazioni che pretende il massimo del realismo, sia quelle del comune appassionato di videogame giocabili e non eccessivamente complicati. Il primo potrà apprezzare la già citata verosimiglianza di ogni aspetto del gioco con le situazioni reali, al secondo non dispiacerà la coreografia che accompagna il tutto.

La grafica non è eccezionale ma è sempre difficile riscontrare tale qualità in una simulazione di volo. Musica ed effetti sonori buoni.

• **Game and Match (set 2).** Si tratta di una raccolta di 10 fra i maggiori successi lanciati sul mercato dalla Ocean. I giochi sono completamente indipendenti uno dall'altro, cioè ognuno di essi può essere caricato in qualsiasi momento senza passare per un menù iniziale comune; non esistendo un punteggio totale che tenga conto degli esiti dei singoli giochi non è possibile compiere gare o confrontarsi con record già stabiliti. Vediamo in dettaglio quali sport sono stati scelti per formare questa raccolta:

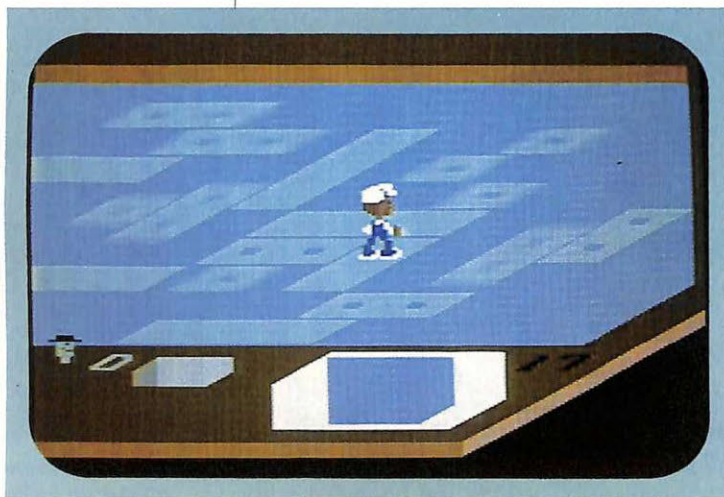
- *Basket Master:* partita di pallacanestro contro il computer o contro un vero avversario. Note particolari: barra di energia che indica lo stato fisico del giocatore, indicatore di posizione ideale per rubare la palla, conteggio dei falli e perdita automatica se se ne commettono troppi.
- *Championship Sprint:* gara automobilistica con possibilità di scelta fra otto piste diverse o costruzione di una propria. Note particolari: bonus durante il percorso, menù scorrevoli a scomparsa, ostacoli come olio, acqua e ghiaccio.
- *Iam Botham's Test Match:* partita di cricket. Note particolari: possibilità di schierare i giocatori nella posizione desiderata, ampia gamma di colpi di battuta, quattro tipi di lanci della palla.
- *Match Day II:* partita di calcio contro il computer, contro un vero avversario o contro il computer giocando in due. Note particolari: selezione delle tattiche, scelta della potenza di tiro, possibilità di disputare un campionato o una coppa di lega.
- *Nick Faldos Open:* torneo di golf a 18 buche. Note particolari: ampia scelta delle mazze e quindi della potenza del tiro, indicatore del vento, visualizzazione della mappa del green, indicazione del numero di colpi in cui si dovrebbe mettere la palla in buca (par).
- *Steve Davis Snooker:* biliardo all'inglese contro il computer o contro un vero avversario. Le biglie sono 21 (15 rosse, 6 di diversi colori). Il giocatore deve mettere in buca nell'ordine una palla rossa (1 punto) seguita da una palla colorata (più punti) fino ad esaurimento delle biglie rosse (quelle colorate vanno sempre rimesse sul tavolo).
- *Superbowl:* finale del campionato professionistico di football americano. Note particolari: possibilità di scelta della potenza dei lanci e dei calci, delle azioni e delle tattiche. I movimenti di un'azione selezionata possono essere visti prima che essa venga giocata.
- *Super Hang on:* gara motociclistica in quattro continenti. Note particolari: accensione del turbo oltre i 280 km/h, tempo massimo per completare le varie fasi, numero di fasi (e quindi difficoltà) variabile da continente a continente.
- *Track & Field:* sei gare di atletica leg-

Game and Match (2). L'immagine sulla confezione della bella raccolta di giochi sportivi.

gera (100 metri, salto in lungo, lancio del giavellotto, 110 metri ostacoli, lancio del martello, salto in alto). Note particolari: tempi o distanze di qualificazione, possibilità di competere in una sola gara o in tutte.

- *Winter Olympiad:* olimpiadi invernali comprendenti salto dal trampolino, discesa libera, bob a due, slalom. Note particolari: possibilità di scelta da 1 a 6 giocatori, gare selezionabili tutte all'inizio, cerimonia di apertura.

Nel complesso Game and Match presenta un'ottima giocabilità grazie, ovviamente, alla quantità di giochi di cui dispone, anche se non tutti i programmi sono dello stesso livello qualitativo, soprattutto dal punto di vista grafico.

• **Compilation sportiva.** Passiamo in casa Epix per vedere un'altra compilation di genere leggermente diverso; si tratta infatti sempre di sport ma con una caratteristica che li accomuna: il filo conduttore che collega le discipline che compongono la raccolta è la particolarità di essere sport o passatempi caratteristici di una determinata nazione; così potremo osservare gare di sollevamento pesi (Russia), di salto del barile (Germania), di tuffo dallo scoglio (Messico), di slalom (Francia), di corsa sui tronchi (Canada), di lancio del palo (Scozia), di sumo (Giappone) e, per finire, un rodeo (Stati Uniti).

L'esito di ogni gioco contribuisce o meno ad arricchire il medagliere di ogni giocatore: più la medaglia è pregiata maggiore è il punteggio che ad essa corrisponde, il miglior totale alla fine di tutte le gare determina il vincitore.

All'inizio del gioco ogni partecipante può scrivere il suo nome e scegliere la nazione da rappresentare, secondo la già consolidata prassi di molti giochi sportivi di casa Epix.

La qualità dei giochi è mediamente non elevatissima, soprattutto a causa della semplicità dei singoli sport e quindi della loro ripetitività. In ogni caso è possibile ovviare a quest'ultimo aspetto selezionando l'opzione che permette di gareggiare in tutte le specialità consecutivamente. Volendo fare esercizio in una singola disciplina (cosa non sempre necessaria vista l'immediatezza di alcuni giochi) potrete usufruire dell'apposita opzione di allenamento.

La grafica è piuttosto elementare, la musica discreta. Da menzionare le brevi introduzioni che appaiono ogni volta che viene caricata una disciplina e che ne spiegano le origini e le regole.

Rock'n Bolt. Il gioco, piuttosto originale come strategia, è caratterizzato da una grafica non troppo sofisticata ma da un'ottima giocabilità.

• **Rock'n'bolt,** originale gioco della Mastertronic. Il vostro compito è quello di costruire un grattacielo di 100 piani, assemblando le travi fra loro e imbullonandole in più punti. Finché non sono saldate alla struttura le travi oscillano continuamente rendendo difficili gli spostamenti dell'operaio. I bulloni possono essere sia avvitati che svitati poiché spesso capita di collegare un certo numero di elementi in una posizione tale da non permetterne l'unione con gli altri. In questo caso non resta che disfare una parte del lavoro e cercare un'altra soluzione.

I quadri da completare (100) sono di due tipi: alcuni lasciano piena libertà sul modo di collegare le travi mentre altri presentano uno schema già stabilito al quale bisogna sottostare. Il punteggio si incrementa quando si avvita e diminuisce quando si svita un bullone; in caso di superamento del tempo limite si perde una vita. Rock'n'bolt è apprezzabile soprattutto dal punto di vista della giocabilità grazie all'idea originale dalla quale prende spunto. Peccato che ad essa non corrisponda un'adeguata sezione grafica, troppo elementare e poco curata. Da menzionare i bonus che è possibile raccogliere durante la prova (punti extra e vite supplementari).

Gianni Arioli
Massimiliano Del Rio

- 80 COLONNE A 10 CPI
- 130 CPS IN ALTA VELOCITÀ
- 24 CPS IN ALTA DEFINIZIONE
- COLLEGABILE A HOME E PERSONAL COMPUTER
- MASSIMA SILENZIOSITÀ

Puoi trovare la MT 81 in tutta Italia presso le reti di vendita di: ***MILANO*** *- SILVERSTAR LTD SPA - TEL. 02/4996* ***MILANO*** *- ACS ELETTRONICA SPA - TEL. 02/5398721-5694082* ***MILANO*** *- CLAITRON SPA - TEL. 02/3010091* ***TORINO*** *- ABACUS SRL - TEL. 011/6680164* ***VERONA*** *- TELESTORE 2 SRL - TEL. 045/541051* ***TRIESTE*** *- I.B.C. SRL - TEL. 040/733395* ***REGGIO EMILIA*** *- H.S.D. SRL - TEL. 0522/557600* ***BOLOGNA*** *- NON STOP SPA - TEL. 051/765299* ***RAVENNA*** *- S.H.R. ITALIA SRL - TEL. 0544/463200* ***FIRENZE*** *- DEDO SISTEMI SPA - TEL. 055/4360251-4361901-4361902* ***ROMA*** *- ALTEC SRL - TEL. 06/3605943-3615744-3615745* ***NAPOLI*** *- MASTERS INFORMATICA SRL - TEL. 081/7703024-7703025* ***PALERMO*** *- BELCO SRL - TEL. 091/547566-545827*

MANNESMANN
TALLY
Stampanti in assoluto

MANNESMANN TALLY srl
20094 Corsico (MI)
Via Borsini, 6
Tel. (02) 4502850/55/60/65/70

adverteam

Effetti speciali professionali

Continuiamo la serie dedicata alle applicazioni software Midi sul C64, basate sull'utilizzo di semplici utility che, in mancanza di sofisticate master keyboards o Midi box nel vostro sistema Midi , possono aiutarvi nella realizzazione di quegli effetti speciali che finora avete sempre invidiato ai professionisti.

Se vi sono piaciute le routine sperimentali pubblicate nella precedente puntata, questa volta apprezzerete la routine che presentiamo, che utilizza ancora l'adattatore per cinque pedali esaminato nello stesso articolo, ma con prestazioni ancora più sofisticate.

Il programma in questione, chiamato semplicemente Pedali, consente di assegnare a video per ciascun pedale, nel classico modo user friendly, il numero della nota da suonare, il canale MIDI su cui la nota verrà suonata e la relativa velocità.

Inoltre il programma permette di assegnare al quinto pedale una nota anche sul rilascio, in modo da ottenere, utilizzando come expander una batteria elettronica, il famoso effetto hit-hat o, a seconda dei vostri gusti, qualunque altro abbinamento anche tramite un qualunque expander o campionatore.

In questo caso precisiamo che la durata della nota al rilascio del pedale è determinata da un ciclo For-Next da 1 a 200.

Se volete sperimentare durate diverse, listate la linea 1140 e cambiate questo valore.

Funzionamento e uso

Dato il Run, il menù presenta due opzioni: Play e Edit. Premendo F1 si entra nel menù di Edit, in cui il programma chiede di inserire sei numeri per le sei note e sei numeri per i relativi canali MIDI e velocità. Esaurita questa operazione si torna automaticamente al menù iniziale.

Ora si può premere F3 per entrare nella funzione Play. Qui verranno visualizzate in una tabella le scelte operate nella precedente sezione; a questo punto il computer è pronto per dirigere la tastiera, l'expander o la drum machine, in funzione ovviamente del canale su cui tali apparati verranno predisposti.

Inserite l'adattatore che avete realizzato secondo le istruzioni fornite nell'articolo precedente nella porta 2 del joystick e, premendo uno dei cinque pedali che avrete collegato all'adattatore, potrete udire le note relative. Nell'angolo in alto dello schermo, ogni volta che premerete un pedale, sarà visualizzato un numero, che scomparirà al rilascio del pedale e che indica il numero della nota suonata.

Quando vorrete cambiare i valori scelti in precedenza, sarà sufficiente premere contemporaneamente il primo e il quinto pedale per tornare al menù principale. Questa operazione può essere compiuta anche con il joystick premendo il tasto Fire e, contemporaneamente, muovendo la barra in direzione Nord.

Analizzando il listato del programma potrete rendervi conto dell'effettiva facilità di comunicazione computer-mondo MIDI.

Resta inteso, come già sottolineato, che questi programmi funzionano con l'interfaccia per C64 pubblicata da Beppe Brigatti su questa stessa rivista (*RE&C* 6/1987). Qualora i lettori dispongano di altre interfacce, dovranno correggere i relativi indirizzi per i registri di stato e dei dati nei listati.

Come da listato, i due indirizzi da noi usati sono rispettivamente 56832 per il registro di stato e 56833 per il registro dei dati.

In una precedente puntata (*RE&C* 10/1988) abbiamo pubblicato una tabella in cui sono indicati gli indirizzi di altre interfacce MIDI.

Consigliamo inoltre di fare riferimento alla puntata apparsa sul numero scorso per quanto riguarda il tipo di pulsante presente sul pedale.

A questo proposito precisiamo che, per consentire l'immediata prova dei programmi, i listati sono elaborati per lavorare con pulsanti normalmente aperti, in modo da poter utilizzare eventualmente il joystick al posto dei pedali.

Assegnazione di accordi ai pedali

Dopo aver realizzato l'adattatore per cinque pedali della puntata precedente, nella quale abbiamo considerato una serie di applicazioni verso una drum machine, vediamo come possiamo utilizzarlo in abbinamento a un sintetizzatore o a un expander, ampliando l'esperienza dell'ultima routine.

Sintetizzatore polifonico multitimbrico K - 1/K - 1M Kawai

Questo nuovo sintetizzatore polifonico dispone di ben 256 forme d'onda campionate in PCM, alcune delle quali lunghe più di un secondo. Di queste, nel caso di un suono Single, è possibile usarne quattro contemporaneamente e, in questo caso, il sintetizzatore si comporta come polifonico a 8 voci. Nel caso in cui si facesse uso di due sole voci tra le quattro possibili, il sintetizzatore diviene polifonico a 16 voci.

La tastiera a 61 tasti è di dimensione standard. L'after touch della nota emessa è splittabile in 8 punti.

Il sintetizzatore dispone inoltre di 64 memorie interne per le funzioni in modo Single e di 32 memorie interne per le funzioni in modo Multi. Queste memorie possono essere raddoppiate usando la cartridge DC-8, fino ad ottenere un totale di 192 suoni come massimo.

La programmazione della tastiera sintetizzatore è particolarmente facile, studiata soprattutto per chi non ha intenzione di approfondire la conoscenza delle complesse tecniche di sintesi. Malgrado ciò lo strumento offre illimitate variazioni sonore.

Questo apparato Midi è distribuito in Italia da: C.B. Music srl, Via Padova 30, 20127 Milano, tel. 02/2895022.

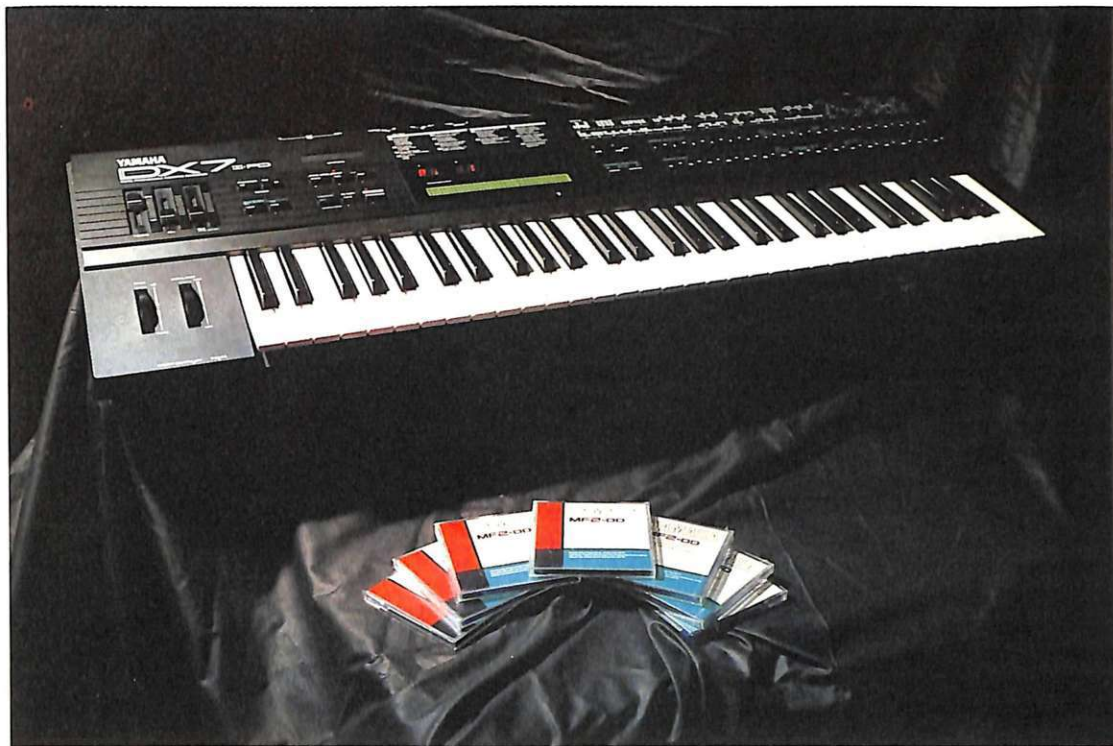
Tabella di implementazione MIDI

[Digital Programmable Algorithm Synthesizer] Date : 12/24, 1986
Model DX7-2 MIDI Implementation Chart Version : 1.0

Function ...		Transmitted	Recognized	Remarks
Basic Channel	Default	1 - 16	1 - 16	memorized
	Changed	1 - 16	1 - 16	
Mode	Default	3	1, 2, 3, 4	memorized
	Messages	x	POLY, MONO(M=1)	
	Altered	xxxxxxxxxxxxxx	x *2	
Note Number		36 - 96 *1	0 - 127 *2	
	True voice	xxxxxxxxxxxxxx	1 - 127	
Velocity	Note ON	o 9nH,v=1-127	o v=1-127	
	Note OFF	x 9nH,v=0	x	
After Touch	Key's	x	x	
	Ch's	o *1	o *2	
Pitch Bender		o *1	o 0-12 semi *2	7 bit resolution
Control Change	1	o *1	o *2	Modulation wheel
	2	o *1	o *2	Breath control
	4	o *1	o *2	Foot Controller
	5	x	o *2	Portamento time
	7	o *1	o *2	Volume
	8	x	o *2	Balance
	10	x	o *2	Pan
	64	o *1	o *2	Sustain foot sw
	65	o *1	o *2	Portamento f sw
	66	o *1	o *2	Sostenuto
	67	o *1	o *2	Soft
	5-31	o *1	o (11-31) *2	Continuous slidr
	11-31	x	o *2	MIDI IN control
Prog Change		o 0 - 127 *1	o 0 - 127 *2	
	True #	xxxxxxxxxxxxxx	0 - 127	64-127:Cartridge
System Exclusive		o *3	o *3	Voice parameters
System Common	Song Pos	x	x	
	Song Sel	x	x	
	Tune	x	x	
System Real Time	Clock	x	x	
	Commands	x	x	
Aux Messages	Local ON/OFF	x	x	
	All Notes OFF	x	o (126,127)	
	Active Sense	o	o	
	Reset	x	x	

Notes: *1 = transmit if transmit channel is not off.
*2 = receive if receive channel is not off.
*3 = transmit/receive if device number is not off.

Mode 1 : OMNI ON, POLY Mode 2 : OMNI ON, MONO o : Yes
Mode 3 : OMNI OFF, POLY Mode 4 : OMNI OFF, MONO x : No

Sintetizzatore DX 7 II - FD Yamaha

Supponiamo che abbiate bisogno di eseguire un certo numero di accordi (meglio se minore o uguale a cinque, altrimenti avrete la necessità di aumentare il numero di pedali, collegando i successivi all'altra presa del joystick) durante l'esecuzione, mentre con le mani siete impegnati a eseguire rispettivamente il basso e l'a solo sulla tastiera.

Per mezzo della routine numero 2, chiamata Assegnazione Accordi, è possibile assegnare a video un accordo diverso per ogni pedale, scegliendo per ciascuno di essi le tre note che lo costituiscono e, globalmente per ciascun accordo, il valore di velocity per le note e il numero del canale MIDI.

Una piccola spiegazione anche per questa routine: come con il programma precedente, appena dato il Run appare il menù principale; entrati nella fase di Edit, vengono richiesti per ciascun pedale i tre numeri relativi alle tre note dell'accordo, il numero del canale MIDI e la velocity dell'accordo.

Per il resto questa routine si comporta similmente alla precedente, tranne naturalmente per il fatto che, ogni volta che verrà premuto uno dei pedali (che andranno collegati alla presa numero 2 del joystick tramite l'adattattore descritto nella puntata precedente), il risultato sarà un accordo anziché una singola nota.

Essendo il listato molto semplice, vi sarà immediato capirne il funzionamento ed eventualmente modificarlo a vostro piacere.

Un consiglio: se volete potete aumentare il numero di note per accordo, ma tenete presente che il Basic, data la sua estrema lentezza, non si presta bene alla trasmissione di messaggi di note che dovrebbero essere generate simultaneamente, quali gli accordi.

Se infatti proverete a mandare molte note per volta, l'unico risultato sarà un arpeggio di note molto ravvicinate tra loro, ma non un accordo.

Questo risultato potrebbe comunque essere un effetto piacevole da sperimentare, sebbene occorra apportare qualche accorgimento al programma per far sì che l'arpeggio sia a tempo con le vostre esecuzioni. Ne parleremo nella prossima puntata. Nel frattempo, osservando le prestazioni di queste utility, vi verrà senz'altro in mente qualche variante per soddisfare certe vostre particolari esigenze pratiche.

Se avete seguito con attenzione fin qui le nostre puntate, ormai non dovrebbe esservi difficile cimentarvi da soli.

(continua)

Cinque Run per cinque urrà!

I trucchi di questo mese sono del tutto speciali: cinque microutility di grande valore di cui non potrete fare a meno non appena capirete di che cosa si tratta. Ci sono addirittura una mini espansione del Basic e un'utility di protezione per i dischi, il tutto in pochi byte di programma.

New irq

Poter modificare il puntatore alla routine di interrupt significa avere a disposizione un utilissimo e potentissimo strumento per realizzare effetti incredibili. Osservando attentamente un videogame dell'ultima generazione si finisce sempre per porsi una domanda di questo tipo: come è possibile che il C64 riesca a fare tutte quelle cose contemporaneamente? Lo scenario del gioco scorre velocemente in una direzione, mentre decine di sprite si muovono quasi come se fossero dotati di volontà autonoma, in direzioni diverse a velocità impressionante. Il trucco sta proprio nella manipolazione della routine di interrupt. New irq è una routine in linguaggio macchina che vi permetterà di modificare il puntatore alla routine di interrupt, noto anche come vettore di interrupt, per fare in modo che punti a una vostra routine e non più a quella di sistema. Per utilizzare la routine dovete copiare il **listato 1** e quindi digitare il consueto Run. Quando riappare il Ready del Basic dovete caricare in memoria la vostra routine in linguaggio macchina che volete eseguire da interrupt e quindi digitare:

Sys 828, h, l

I parametri h e l rappresentano rispettivamente il byte alto e basso dell'indirizzo di inizio della vostra routine. Per calcolare i valori di questi parametri basta usare la formula:

h=int(a/256): l=a-h*256

dove a rappresenta l'indirizzo di inizio della vostra routine.

Listato 1. New IRQ

```
10 rem -----
20 rem -          -
30 rem - new irq -
40 rem -          -
50 rem -----
60 :
70 fori=828to848:reada:ck=ck+a:pokei,a:next
80 ifck<>2299 then print"<CLEAR>errore nei data":end
90 :
100 rem sys 828, h, l per attivare
110 rem la routine
120 rem h=byte alto
130 rem l=byte basso
140 rem della nuova routine di
150 rem interrupt
160 :
170 data 120,032,253,174
180 data 032,158,183,142
190 data 021,003,032,253
200 data 174,032,158,183
210 data 142,020,003,088
220 data 096
```

Multi delete

Quando si programma in Basic capita spesso di dover cancellare molte linee di programma consecutive. In situazioni del genere la procedura che si segue normalmente consiste nel digitare i numeri di tutte le linee da cancellare premendo Return dopo ciascuno di essi. Con Multi delete potrete finalmente svolgere più velocemente e con molta meno fatica questa operazione. La routine è in grado di cancellare un gruppo di linee Basic consecutive: vi basterà indi-

Listato 2. Multi delete

```
1 rem ————————
2 rem -            -
3 rem - multi delete -
4 rem -            -
5 rem ————————
6 :
20 rem *** le linee (20-110)
30 rem *** saranno cancellate
40 rem *** digitando run 60000
50 rem ***
60 rem ***
70 rem ***
80 rem ***
90 rem ***
100 rem ***
110 rem ***
60000 i1=10: rem (prima linea - incremento fra le linee )
60001 i1=i1+10: rem (incremento)
60002 if i1=120 then end: rem (ultima linea da cancellare)
60003 print chr$(147):print:print
60004 printi1:print"i1="i1":goto 60001":print chr$(19)
60005 for i2=631 to 633:poke i2,13:next:poke198,3
```

Listato 3. True restore

```
1 rem ————————
2 rem -              -
3 rem - true restore -
4 rem -              -
5 rem ————————
6 :
10 sum=0:fora=2816to2885:readb:sum=sum+b
20 poke a,b:next
30 if sum<>6299 then print"<CLEAR>errore nei data":end
40 sys 2816
50 data 169,14,141,0
60 data 10,169,11,141
70 data 1,10,32,37
80 data 11,96,32,37
90 data 11,173,70,11
100 data 141,33,208,173
110 data 71,11,141,32
120 data 208,173,72,11
130 data 133,241,76,3
140 data 64,120,169,50
150 data 141,24,3,169
160 data 11,141,25,3
170 data 88,96,173,32
180 data 208,141,71,11
190 data 173,33,208,141
200 data 70,11,165,241
210 data 141,72,11,76
220 data 64,250
```

care la prima e l'ultima linea del gruppo da rimuovere e Multi delete farà il resto, velocemente e senza possibilità di errore. La routine è scritta interamente in Basic e per poterla usare dovrete copiare il **listato 2** e agganciarlo al vostro programma Basic. Quando dovete cancellare una serie di linee contigue, non dovete far altro che assegnare alla variabile i1, che compare nella linea 60000, il numero corrispondente alla prima linea da cancellare (l'incremento fra le linee da cancellare); sostituire al valore numerico che compare nella linea 60001 il valore corrispondente all'incremento fra le linee da cancellare (se volete cancellare tutte le linee comprese in un certo range dovete inserire il valore 1); sostituire il valore che compare nella linea 60002 con il numero corrispondente all'ultima linea da cancellare più l'incremento stesso. Per esempio, per cancellare le linee 20, 30 e 40 di un programma, dovrete inserire nell'ordine i seguenti valori: 10, 10, 50.

True restore

Quando si usa la combinazione di tasti Run/Stop e Restore per resettare il Commodore 64, oltre all'effetto desiderato si ottiene anche un indesiderato effetto collaterale: il colore del bordo e del fondo dello schermo vengono infatti cambiati. Se volete eliminare questo fastidio potete usare True restore. La routine è scritta completamente in linguaggio macchina e per poterla usare dovete copiare il **listato 3** e digitare il consueto Run. Il caricatore Basic provvede automaticamente ad attivare True restore quindi, quando compare il Ready del Basic, la routine è già installata correttamente in memoria.

Listato 4. Disk Header

```
1 rem ———————
2 rem -             -
3 rem - disk header -
4 rem -             -
5 rem ———————
6 :
50 close 7:close 8:open7,8,15,"io"
60 s=144:open8,8,8,"#"
100 poke 198,0:printchr$(147);
101 printchr$(18)tab(15)"disk header"
102 print:print" i-nvisible directory"
104 print:print" r-ename disk header"
105 print:print" c-ustom header id"
108 print:print"     i, r, or c ?"
109 gosub 200
110 geta$:ifa$><"i"anda$><"r"anda$<>"c"then110
115 ifa$="i"thenford=1to3:d$=d$+chr$(0):next:l=3
120 ifa$="c"theninput"new id";d$:s=162:l=5
125 ifa$="r"theninput"new header";d$:l=16
130 ifa$<>"i"thenford=len(d$)tol:d$=d$+chr$(160):next
160 print#7,"b-p:";8;s:print#8,left$(d$,l);
170 print#7,"u2:";8;0;18;0:gosub200
180 close 8:print#7,"io":close7:stop
200 print:print#7,"u1:"8;0;18;0
210 print#7,"b-p:";8;144:printchr$(18);
220 ford=1to23:get#8,a$:a=asc(a$+" ")
230 printchr$(a);:next:print:print:return
```

Disk header

Questa micro utility vi permetterà di modificare l'ID e il nome di un disco, nonché di rendere illegibile la directory (con l'usuale procedura Load"$", 8 e List). Può quindi essere usata per realizzare una mini protezione del vostro software più prezioso. La routine fa uso dei comandi per l'accesso diretto al disco, e quindi può esservi utile leggerla con attenzione per comprendere meglio il funzionamento di questi comandi.

Per utilizzare Disk header copiate il **listato 4** e date il consueto Run. Verrà visualizzato un me-

nù con le seguenti opzioni, attivabili premendo il tasto indicato all'estrema sinistra dell'opzione :

- **I-nvisible directory**: permette di rendere illeggibile, con la solita procedura Load"$", 8 e List, la directory del disco. Per rendere nuovamente leggibile la directory basta riselezionare questa opzione.
- **R-ename disk header**: permette di cambiare il nome del disco nel drive. Selezionata l'opzione dovete inserire il nome da assegnare al disco e premere Return.
- **C-ustom header ID**: permette di modificare l'ID del disco nel drive. Selezionata l'opzione dovete inserire i cinque caratteri che costituiranno il nuovo ID del disco.

Basic tracer

Si tratta di una vera e propria espansione del Basic. Basic tracer aggiunge al Basic standard un nuovo comando, un asterisco (*), che vi permetterà di tracciare l'esecuzione del programma Basic in memoria. Questa mini espansione dovrà sempre essere nella memoria del C64, perché è davvero utilissima e non ruba neanche un byte alla RAM Basic. Basic tracer è scritta interamente in linguaggio macchina e per utilizzarla dovete copiare il **listato 5**. Per tracciare l'esecuzione del programma Basic in memoria non dovete far altro che digitare l'asterisco (*) e premere Return. Per disattivare il tracciamento dovete premere nuovamente la stessa sequenza di tasti. La routine è insensibile ai reset e può essere disattivata solo spegnendo il computer.

Daniele Maggio

Listato 5. Basic Tracer

```
1 rem ————————
2 rem -              -
3 rem - basic tracer -
4 rem -              -
5 rem ————————
6 :
10 ad=679:msg=420858192*(peek(65532)=34):q$=chr$(34)
11 readdt:ifdt<0thenon(ad=764)+2goto14,15
12 ifdt>255thenon(ck=dt)+2goto11,15
13 ck=ck+dt:pokead,dt:ad=ad+1:goto11
14 sys 700:end
15 poke57,peek(63):poke58,peek(64):print"?datastatement";:sysmsg
679 data 32,115,0,208
700 data 12,174,8,3
701 data 172,187,2,142
702 data 187,2,140,8
703 data 3,76,174,167
704 data 224,2036,32,237
705 data 255,224,22,208
706 data 18,169,199,141
707 data 186,2,141,251
708 data 2,169,203,4495
709 data 141,238,2,169
710 data 221,141,233,2
711 data 162,239,160,2
712 data 142,8,3,140
713 data 9,3,96,6606
714 data 164,58,200,240
715 data 10,230,199,32
716 data 201,189,198,199
717 data 32,59,171,32
718 data 115,0,8935
719 data 201,172,240,177
720 data 32,121,0,76
721 data 231,167,10352,-1
```

Per trasferire i programmi di RE&C

Molti lettori hanno incontrato difficoltà nell'eseguire le operazioni di trasferimento dei programmi da nastro a disco. L'utility Dsave richiedeva infatti di specificare due indirizzi che definivano la zona occupata dal programma da trasferire. La sequenza di operazioni non era sempre uguale per tutti i programmi, perciò molti lettori inesperti non hanno potuto operare tutti i trasferimenti desiderati. In questo numero è disponibile sulla cassetta allegata una nuova utility denominata Dsave v2.

Dopo il caricamento del solito menù all'inizio della cassetta, tutti coloro che vorranno trasferire i programmi sul disco dovranno caricare e lanciare Dsave v2. Il menù offre tre possibilità:

1 - La cassetta verrà letta e il primo programma incontrato caricato. A questo punto viene chiesta conferma per il trasferimento sul disco, dopodiché si passerà al caricamento del successivo programma sulla cassetta e cosi via.

2 - Scegliendo la seconda opzione, invece, verranno salvati su disco tutti i programmi automaticamente, senza selezioni da parte dell'utente. A questo proposito bisogna ricordare che, a volte, sulla cassetta vi sono dei file sequenziali o programmi particolari che il copiatore non può trasferire, nel qual caso il processo si blocca. Vi consigliamo quindi di utilizzare l'opzione 1 anche se volete trasferire tutti i programmi della cassetta.

3 - Questa opzione consente di visionare la directory del disco.

Integrali: esercizi al computer

In questo numero vi abbiamo preparato un simpatico programma che vi permetterà, pur facendovi divertire, di acquisire quella padronanza indispensabile a chi vuole risolvere qualsiasi problema del calcolo integrale.

Abbiamo deciso di fare una piccola pausa per quanto riguarda la teoria, per darvi modo di portarvi a pari con i vostri programmi scolastici e per darvi la possibilità di iniziare a cimentarvi con la risoluzione di integrali indefiniti.

Abbiamo già visto che cosa si intende per integrale indefinito: si tratta in pratica di definire una funzione che, derivata una volta nella variabile x, fornisca l'equazione della funzione integranda.

Si tratta quindi di una ricerca di primitive e, in definitiva, occorre fare il ragionamento opposto rispetto a quello che fate per calcolare la derivata di una funzione. Mentre prima la linea logica da seguire era "trovare la derivata della funzione ...", ora sarà "trovare la funzione che derivata è ...".

La prima cosa che viene alla mente, quando si deve risolvere un problema di questo tipo, è quindi quella di cercare una funzione conosciuta che soddisfi opportune condizioni e che, opportunamente derivata, risolva il problema.

Anche se questa sembrerebbe la linea più logica, potete facilmente rendervi conto che raramente il problema è di così facile soluzione e proprio per questo sono stati elaborati dei metodi per facilitare il compito. Innanzi tutto esistono delle funzioni di cui conosciamo la derivata in quanto di facile calcolo: in questo caso la soluzione dell'integrale è davvero banale. Ci sono poi altre regole ben precise di derivazione (prodotto, somma, funzione di funzione, eccetera) che ci aiutano a individuare la via da seguire.

Per maggior chiarezza, in **tavola 1** sono riportati tutti i casi di integrali particolarmente semplici che proprio per questo vengono definiti immediati. Vedete subito che, se leggete la tabella da destra a sinistra, ottenete l'elenco delle derivate fondamentali.

L'argomento di teoria sarà comunque sviscerato nel numero prossimo, dove vi verranno mostrate in dettaglio tutte le tecniche per risolvere anche i più difficili integrali.

Per questa puntata vi proponiamo un programma eserciziario che vi metterà a dura prova e metterà in luce tutti i buchi della vostra preparazione matematica.

Come livello di difficoltà, è un lavoro destinato a studenti delle scuole medie superiori (in particolare licei scientifici e istituti tecnici) ma che, per la vastità degli argomenti trattati, bene si presta come allenamento anche per studenti universitari impegnati in problemi di Analisi Matematica 1. Si tratta in pratica di un test in cui il candidato predetermina sia il numero di esercizi sia la durata della prova, e in cui alla fine ha la possibilità di verificare se quanto ha fatto è corretto.

Per capire esattamente di che cosa si tratta, caricate il programma Integrali con la solita procedura e lanciatelo (Run). La maschera di input vi spiegherà che

Una schermata del programma Eserciziario.

cosa riesce a fare il programma. Selezionate per prima cosa il numero di esercizi (da un minimo di 5 a un massimo di 30) e il tempo da impiegare (da 15 a 150 minuti). Queste due condizioni sono prioritarie e non è possibile iniziare la prova senza fornirle alla macchina. In ogni caso il computer vi avviserà se si verifica una qualsiasi mancanza.

Tenete sempre presente che non è possibile interrompere una prova durante il suo svolgimento senza perdere tutti i dati introdotti: fate perciò bene attenzione al tempo che avete a disposizione prima di iniziare. Noterete che prima di iniziare la prova vi viene suggerito di mettere su carta sia il testo sia la soluzione dell'esercizio: questo al fine di facilitarvi la fase di correzione, in quanto la soluzione spesso può apparire di difficile interpretazione.

Nella pagina seguente vi viene chiesto se desiderate utilizzare il nastro o il disco: ogni volta che digitate la soluzione di un esercizio, questa viene riportata su memoria di massa per poter essere successivamente utilizzata dal programma Analisi che vi fornirà la correzione. Dovete quindi decidere se usare disco o nastro, anche se in termini di tempo le due soluzioni sono pressoché equivalenti. A questo punto c'è una piccola pausa necessaria al computer per aprire il file (denominato A1), al termine della quale si inizia la fase di calcolo.

Per la ricerca degli esercizi, il programma ha accesso a una banca dati fornita di 100 esercizi tutti diversi, che riuniscono tutte le principali tecniche di calcolo conosciute.

Gli esercizi sono scelti in modo casuale e quindi è molto probabile che anche dopo parecchie volte che utilizzate questo programma si verifichino situazioni del tutto nuove.

Vi invitiamo, in ogni caso, a non utilizzare le precedenti esperienze; ciò significa che, anche se vi capita un esercizio che avete già risolto, cercate di considerarlo come nuovo senza consultare le precedenti esperienze.

Solo in questo modo potrete acquisire padronanza nella risoluzione degli esercizi e vedrete risultati notevolissimi anche in pochi giorni.

Un altro consiglio: abbiamo notato che molto spesso chi si trova a usare questo programma viene preso dall'ansia di fare presto per non utilizzare più tempo di quello selezionato in partenza: questo è un errore che non dovete commettere!

Innanzi tutto, anche se andate oltre il tempo consentito, il programma vi permette comunque di arrivare fino in fondo alla prova. Secondariamente impiegare un minuto in meno e poi sbagliare un esercizio è controproducente. Ciò non significa che potete impiegare ore, però non affannatevi e controllate bene la soluzione prima di confermarla.

Una volta risolto l'esercizio, potete inserire la soluzione nell'apposito spazio tenendo presente la sintassi del Commodore 64. L'unica funzione di cui avrete bi-

Tavola 1 - Integrali fondamentali

I. $\int x^n\,dx = \frac{x^{n+1}}{n+1} + C,\ n \neq -1.$

II. $\int \frac{dx}{x} = \ln|x| + C.$

III. $\int \frac{dx}{x^2+a^2} = \frac{1}{a}\operatorname{arctg}\frac{x}{a} + C = -\frac{1}{a}\operatorname{arcctg}\frac{x}{a} + C_1 \quad (a \neq 0).$

IV. $\int \frac{dx}{x^2-a^2} = \frac{1}{2a}\ln\left|\frac{x-a}{x+a}\right| + C \quad (a \neq 0).$

$\int \frac{dx}{a^2-x^2} = \frac{1}{2a}\ln \times \left|\frac{a+x}{a-x}\right| + C \quad (a \neq 0).$

V. $\int \frac{dx}{\sqrt{x^2+a}} = \ln|x + \sqrt{x^2+a}| + C \quad (a \neq 0).$

VI. $\int \frac{dx}{\sqrt{a^2-x^2}} = \operatorname{arcsen}\frac{x}{a} + C = -\arccos\frac{x}{a} + C \quad (a > 0).$

VII. $\int a^x\,dx = \frac{a^x}{\ln a} + C \quad (a > 0);$

$\int e^x\,dx = e^x + C.$

VIII. $\int \operatorname{sen} x\,dx = -\cos x + C.$

IX. $\int \cos x\,dx = \operatorname{sen} x + C.$

X. $\int \frac{dx}{\cos^2 x} = \operatorname{tg} x + C.$

XI. $\int \frac{dx}{\operatorname{sen}^2 x} = -\operatorname{ctg} x + C.$

XII. $\int \frac{dx}{\operatorname{sen} x} = \ln\left|\operatorname{tg}\frac{x}{2}\right| + C = \ln|\operatorname{cosec} x - \operatorname{ctg} x| + C.$

XIII. $\int \frac{dx}{\cos x} = \ln \left|\operatorname{tg}\left(\frac{x}{2}+\frac{\pi}{4}\right)\right| + C = \ln|\operatorname{tg} x + \sec x| + C.$

XIV. $\int \operatorname{sh} x\,dx = \operatorname{ch} x + C.$

XV. $\int \operatorname{ch} x\,dx = \operatorname{sh} x + C.$

XVI. $\int \frac{dx}{\operatorname{ch}^2 x} = \operatorname{th} x + C.$

XVII. $\int \frac{dx}{\operatorname{sh}^2 x} = -\operatorname{cth} x + C.$

Tavola 2 - Funzioni

FUNZIONI MATEMATICHE	ABBREVIAZIONI COMMODORE
SENO	SIN
COSENO	COS
TANGENTE	TAN
ESPONENZIALE	EXP
VALORE ASSOLUTO	ABS
RADICE QUADRATA	SQR
LOGARITMO NATURALE	LOG
ARCOTANGENTE	ATN

sogno per la risoluzione degli esercizi che non è direttamente codificata dal 64 è l'arcseno, che tuttavia potrete inserire come arcsin. Tutti gli argomenti delle funzioni vanno tra parentesi ed è obbligatorio l'uso degli operatori di moltiplicazione, i quali invece nella scrittura ordinaria molto spesso si omettono. Dovete cioè tenere presente che la linea che state inserendo ha le proprietà di una vera e propria linea di programma e perciò deve essere eseguibile come tale, fatta eccezione per la funzione arcoseno. In **tavola 2** sono riportate tutte le abbreviazioni usate per indicare le funzioni matematiche.

Ricordate anche che ci sono delle priorità ben precise che la macchina rispetta e che dovete tenere presente per essere sicuri che ciò che avete scritto è realmente ciò che volevate scrivere!

In **tavola 3** sono riportate le priorità delle operazioni; in ogni caso prima vengono svolti i calcoli tra parentesi poi le funzioni matematiche con argomento (sin, cos, log eccetera) e poi ^, /, *, -, +.

Fate attenzione a questo, perché la macchina controlla solo la sintassi di ciò che scrivete e vi segnala eventuali errori, ma non può controllare la logica.

Se nell'inserimento della soluzione commettete errori potete correggerli mediante la pressione del tasto Inst/Del. Controllate accuratamente quello che scrivete perché dopo la pressione del tasto Return non potrete più correggerlo se è sintatticamente esatto. Se non conoscete la soluzione premete il tasto Return senza inserire nulla e proseguite.

La macchina calcola per ogni esercizio il tempo che impiegate, e da questo viene detratto anche il tempo impiegato per la battitura della stringa. Non affannatevi perciò sui tasti, perché è tutta fatica inutile. Una volta esauriti gli esercizi, passate alla fase di correzione premendo l'opzione 4: verrà caricato il programma Analisi e anche qui vi verrà chiesto se lo volete caricare da disco o da nastro. Fate in modo che il programma sia già pronto quando sceglierete l'opzione: se dovessero verificarsi errori in questa fase, non preoccupatevi perché non avete perso nulla: caricate Analisi con la normale procedura e fatelo partire. Ora la macchina ha bisogno del file con le vostre risposte: perciò dovete fornirglielo indicando su quale periferica la macchina potrà utilizzarlo.

A questo punto inizia la correzione: per ogni esercizio la macchina vi dirà se la risposta è corretta o meno visualizzando la stringa corrispondente alla vostra soluzione e quella corrispondente a quella fatta da noi. In questo modo potrete verificare le soluzioni, le quali potrebbero essere differenti nella forma (più o meno semplificate) ma non nella sostanza.

Vi verrà quindi fornita un'analisi sui tempi, da cui verificherete la vostra rapidità: il tempo medio per esercizio è riportato in secondi e, per avere un'idea della vostra velocità di calcolo, tenete presente che per un ragazzo delle superiori è stato valutato 200 secondi, mentre per un universitario 170 secondi. Il calcolo è stato fatto su un campione abbastanza significativo di ragazzi, quindi si può ritenere con buona approssimazione che la stima rispetti effettivamente un ipotetico ragazzo medio.

In fase di correzione, nella stringa del computer vedrete la stringa FNAS che sostituisce l'arcoseno, quindi le due espressioni sono equivalenti.

Un'ultima osservazione: nell'introduzione della soluzione nel programma Integrali la lunghezza massima della stringa è di 70 caratteri: se la superarate significa che avete sbagliato, oppure che dovete semplificare l'espressione.

Con questo programma avete un valido strumento per iniziare e per allenarvi nel calcolo integrale, tuttavia il programma è limitato dal fatto di non poter fornire la soluzione per esteso (cioè con tutti i passaggi) di un esercizio che magari non siete riusciti a risolvere.

Nel prossimo numero pubblicheremo un programma che vi permetterà di ottenere la soluzione degli esercizi di questo programma con tutti i passaggi necessari, corredati anche da commenti illustrativi.

Alberto Palazzo

(continua)

Tavola 3 - Precedenze

DA SINISTRA A DESTRA PRIORITÁ DECRESCENTE					
↑ / * - +	POTENZA	DIVISIONE	PRODOTTO	SOTTRAZIONE	SOMMA

Finalmente uno strumento di compattazione completo! Condensazione di ben 250 blocchi di programma, codice flessibile, scompattazione elegante e veloce, unione e compattamento di più file.

File ad alta densità

Quello che vi presentiamo è un mini programma ultra potente che permette di effettuare compattazioni di file e programmi di ogni genere e dimensione.

Gigapack, semplicissimo da usare, riduce le dimensioni di un file lungo fino a 250 blocchi e vi permette di ridurre a un file unico una serie di file. Gigapack è utilissimo per compattare in un programma mono-file routine in linguaggio macchina dislocate nelle più disparate zone della memoria, gestite, magari, da un programma Basic.

Tutte le operazioni per ottenere questi stupefacenti risultati sono molto semplici e si risolvono, al massimo, in un pugno di Peek e Poke.

Chiunque abbia un minimo di confidenza con il Commodore 64 troverà semplice, interessante e istruttivo questo eccezionale programma.

Chi invece avesse comprato il suo primo computer solo da pochi giorni, potrà effettuare semplici compattazioni seguendo passo dopo passo questo articolo.

Compattare un file, come ben sa chi se ne intende, significa elaborare i byte che lo costituiscono in modo da mantenere lo stesso numero di informazioni in uno spazio inferiore. Come ciò venga fatto sarà spiegato in un'altra occasione, ma, grosso modo, si può dire che alcune sequenze di dati del file, per esempio quelle ripetitive, vengono sostituite con byte che ne specificano la caratteristica.

Gigapack

Caricate dalla cassetta, o dal disco su cui lo avete trasferito con Dsave V.2, il programma Gigapack e lanciatelo. Quando lo schermo sarà divenuto nero, e quando il testo in giallo e il messaggio "Gigapack installato!" avranno confermato il corretto funzionamento, inserite il disco o il nastro contenente il file programma da compattare e caricatelo normalmente con il comando Load.

Al termine del caricamento digitate SYS820. Lo schermo si riempirà di caratteri strani e, in basso, comparirà un messaggio.

Questo messaggio indica il tipo di compattazione in atto (sul video dovreste vedere un curioso movimento di simboli vari) e la percentuale di compattamento al termine delle operazioni. A questo proposito, possiamo dire che Gigapack effettua compattamenti veramente efficaci, riducendo almeno del 10 per cento anche programmi già compattati da altri compattatori. La velocità di compattazione e scompattazione è ottima.

Terminata dunque la compattazione, potrete controllare le dimensioni del programma ridotto digitando semplicemente:

```
PRINT PEEK(45) + 256*PEEK(46)
```

Tale operazione, che avreste potuto effettuare anche al termine del caricamento del programma in questione per conoscerne le dimensioni prima del compattamento, va compiuta sempre, in quanto

la funzione di Save che dovrete effettuare funziona correttamente solo per programmi che non superano la locazione 53247. Se quindi il programma compattato non supera i limiti consentiti potete procedere al salvataggio su disco o nastro. Digitate Save seguito dalle virgolette e dal nome del programma aggiungendo ",8" dopo le virgolette di chiusura se salvate su disco.

Il file del programma compattato può essere caricato e lanciato come qualsiasi altro file programma. Dopo il Run si dovrà attendere lo scompattamento del file nelle sue dimensioni originali, operazione che non dura più di 45 secondi per programmi lunghi 250 blocchi.

Durante la compattazione la memoria riservata allo schermo è usata come buffer, evitando interferenze fra Gigapack e il programma in memoria.

Alcune considerazioni

Sicuramente tutti coloro che possiedono il C64 da poco tempo, o comunque che non hanno avuto modo di apprenderne i segreti e i meccanismi di funzionamento più complessi, avranno difficilmente bisogno di compattare programmi più lunghi di 202 blocchi. Il caricamento di tali file, di solito generati da operazioni di manipolazione avanzata, implica molti problemi. Infatti la memoria oltre la locazione 53247 è riservata al sistema operativo e il programma in caricamento, che va necessariamente a sovrapporsi ad esse, le modifica creando spesso il blocco irreversibile del sistema. Gigapack, quando attivato, installa una nuova routine di caricamento che durante l'input disabilita la memoria riservata eliminando ogni problema.

La grande flessibilità d'uso di Gigapack consente anche di compiere qualche artificio. Per esempio si può compattare in un unico file più programmi complementari, ossia programmi costituiti da più moduli generati separatamente: è possibile unire i dati di una schermata in alta risoluzione con la routine per attivarla e la linea Basic per lanciare quest'ultima, oppure unire un programma Basic con tutte le routine in linguaggio macchina che lo implementano.

In sostanza questa utility è lo strumento indispensabile per rendere più professionali le creazioni di chi sa programmare e per risparmiare spazio sui dischi di chi si limita a usare programmi già fatti.

Compattazione di più file

Gigapack effettua un Run al termine dello scompattamento. Perciò è necessario che tutti i vostri moduli in linguaggio macchina vengano messi in azione da una linea Basic caratterizzata da una Sys. Supponiamo che abbiate due routine in linguaggio macchina: la prima rilocata nella RAM nascosta in $F000 (61440), e la seconda che attiva la RAM nascosta e passa l'esecuzione alla routine situata in $C000 (49152), e il cui punto di entrata sia proprio 49152. Per prima cosa attivate Gigapack. Caricate ora la routine situata più in alto (quella in 61440). Al termine esaminate le locazioni 45 e 46 mediante due Peek e prendete nota dei valori che ottenete.

A questo punto digitate NEW e caricate la routine in 49152 digitando ancora NEW dopo il caricamento. Digitate quindi la linea Basic:

10 SYS 49152

e memorizzatela col tasto Return.

A questo punto inserite, con due Poke in 45 e 46, i valori che avete letto prima nelle stesse locazioni, e digitate SYS 820.

Al termine del compattamento salvate il tutto con Save. Il file generato sarà lanciabile come qualsiasi programma.

Benché Gigapack sia un'utility eccezionale e completa, rispetto a quelle del suo genere ha una piccola limitazione: se una routine in linguaggio macchina che desiderate compattare deve girare in una locazione inferiore alla 2048 rischia di interferire con le routine di Gigapack.

Per tale ragione è meglio utilizzare una routine secondaria che, dopo il Run seguente lo scompattamento, riloca la routine in questione, da voi momentaneamente posta altrove, nelle locazioni critiche. In quel momento infatti le routine di Gigapack avranno esaurito la loro funzione.

Raffaele Zanini

Suono-sintesi, Amiga-musica!

Lo stupore per le straordinarie potenzialità di Amiga è da parecchio tempo un'emozione frequente per chi lo possiede che, per quanto ciò possa sembrare strano, non si stupisce più di stupirsi. Alle soddisfazioni di chi utilizza Amiga si contrappone solo l'invidia di chi non ce l'ha.

Non è certamente il caso di parlare ancora una volta delle straordinarie capacità musicali di Amiga. Ricordiamo comunque che Amiga è fornito di quattro canali audio indipendenti e il segnale in uscita è stereo Hi-Fi, con due canali per parte.

Esistono in commercio numerosi programmi musicali che trasformano Amiga in un vero e proprio sintetizzatore musicale: i nostri lettori abituali hanno certamente letto la prova di Sonix presentata nel numero di novembre di *Radio Elettronica & Computer*. A nostro avviso si tratta del programma di sintesi musicale e di sequencing più completo, anche se è possibile che mentre stiamo scrivendo stia già entrando in commercio un software ancora più completo e potente.

Gli esperti in tastiere elettroniche professionali sanno benissimo che esistono due grandi categorie di strumenti: i sintetizzatori veri e propri e i campionatori.

Un sintetizzatore è uno strumento che crea i diversi suoni a partire da un certo numero di parametri, quali il numero di oscillatori, il numero di armoniche, i filtri, i vibrati e così via. In questo modo è possibile produrre una gamma estesissima di suoni. Il programma Sonix sostanzialmente trasforma Amiga in un sintetizzatore.

I campionatori funzionano secondo un principio completamente diverso: tali strumenti campionano un suono reale e ne convertono la frequenza a tutte le note della tastiera.

Supponiamo per esempio che vi interessi suonare con il tintinnìo dei bicchieri. Con un campionatore non dovete fare altro che "fargli ascoltare" il suono prodotto da un bicchiere: i suoi circuiti elettronici provvedono a digitalizzarne la forma d'onda e a registrarla su un

floppy disk. A questo punto basta effettuare la selezione desiderata e il campionatore si trasforma in una fila di bicchieri perfettamente intonata e a vostra disposizione.

Naturalmente non è detto che un campionatore sia migliore o peggiore di un sintetizzatore: un musicista si serve dell'uno o dell'altro a seconda delle proprie necessità, ma è comunque molto comodo disporre di entrambi gli strumenti in modo da potere scegliere di volta in volta l'effetto migliore.

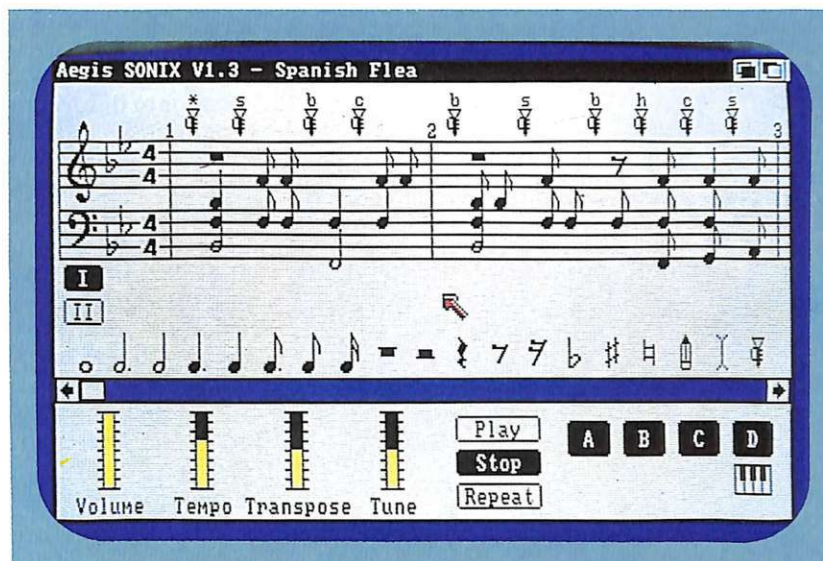
Anche Amiga è in grado di funzionare come un campionatore, però in questo caso è necessario un accessorio che interfacci la sorgente sonora (un microfono o un qualunque apparecchio sonoro) all' hardware di Amiga. Questo accessorio esiste, anzi ne esistono numerosi modelli.

Questo mese abbiamo voluto provarne uno che ci è sembrato particolarmente semplice e, nello stesso tempo, potente ed efficace: Easy Sound.

Come si presenta

L'imballaggio dell'apparecchio è vistosamente sovradimensionato: infatti ha le dimensioni di un grosso libro, mentre Easy Sound è un oggettino piccolo all'incirca come un pacchetto di sigarette. Nella scatola comunque si trovano anche un floppy disk con il software, un microfono e un foglio di istruzioni. Le istruzioni sono solo un misero foglio che spiega ben poco. Anche il microfono non è eccezionale, d'altra parte un buon microfono costa molto più dell'intero Easy Sound.

Spendiamo solo quattro parole per descrivere il software allegato. Si tratta di un buon programma che permette di utilizzare Easy Sound in modo molto semplice, nonostante la scarsità delle istruzioni.

Non ci soffermiamo ulteriormente sul software fornito in dotazione perché Easy Sound è compatibile con il fantastico programma Audiomaster, prodotto dalla Aegis Software, la cui prova è disponibile nella rubrica Software News di questa stessa rivista. Tale programma è infinitamente migliore e consente di sfruttare Easy Sound in modo davvero completo ed efficace; pertanto consigliamo l'acquisto del programma a chiunque intenda acquistare Easy Sound. Del resto è arcinoto che non serve a nulla disporre di un hardware eccezionale se non si dispone di software altrettanto valido.

Ma passiamo alla descrizione di Easy Sound: l'apparecchio è dotato di un connettore per la porta parallela di Amiga 500 o 2000.

Chi possiede un Amiga 1000 deve acquistare anche un apposito adattatore, poiché com'è noto la prima versione di Amiga è dotata di un connettore maschio invece del connettore standard femmina.

Oltre al connettore per Amiga, si trovano due prese per il collegamento con la sorgente sonora: un jack per il microfono e una presa pin per una sorgente attiva. Tra le due prese si trova un controllo di volume che controlla unicamente il microfono.

A nostro avviso l'uso del microfono non è consigliabile, in quanto fornisce un segnale di qualità decisamente peggiore della qualità del campionamento. Se proprio non potete farne a meno, vi consigliamo l'acquisto di un buon microfono con opportuno preamplificatore da collegare alla presa attiva di Easy Sound.

Abbiamo effettuato la prova servendoci di un registratore portatile fornito di un cavetto adatto per collegare l'uscita cuffia con l'entrata attiva di Easy Sound. Naturalmente, poiché Easy Sound è un campionatore monofonico, è possibile collegare solo un canale del registratore.

Esistono in commercio anche campionatori stereofonici che sono in grado di campionare contemporaneamente i canali destro e sinistro e di riprodurre l'effetto stereofonico all'uscita di Amiga.

Tali campionatori possono servire a chi desidera un risultato particolarmente sofisticato, ma presentano l'inevitabile difetto di occupare il doppio della memoria richiesta da un apparecchio mono.

Come funziona

Supponiamo che si voglia campionare un brano musicale qualsiasi.

Si collega l'uscita cuffia dell'apparecchio all'entrata di Easy Sound, si accende l'apparecchio e, incredibile, si sente la musica

uscire dall'altoparlante del monitor di Amiga. La velocità di elaborazione di Amiga è tale da essere in grado di campionare il suono e di riprodurlo contemporaneamente attraverso la propria uscita audio.

Il suono che si sente infatti è il risultato di una conversione analogico-digitale seguita da una conversione digitale-analogica. Questo significa che Easy Sound converte il segnale analogico proveniente dal registratore (l'uso di un registratore appare particolarmente comodo, ma naturalmente è possibile usare un giradischi, una radio o altro) in una sequenza di numeri che viene inviata in tempo reale alla porta parallela di Amiga.

Dopodiché Amiga, grazie al potentissimo chip custom che gestisce l'uscita audio, riconverte la sequenza di numeri in un segnale analogico che viene inviato al monitor.

A questo punto si rende necessario qualche particolare tecnico: un segnale sonoro analogico, quale l'uscita di un registratore, consiste in un segnale elettrico variabile nel tempo con una frequenza minima di 20 Hz e una fequenza massima di 20000 Hz. La tecnica di digitalizzazione sonora consiste nell'effettuare la lettura del livello di tale segnale con una determinata frequenza, detta frequenza di campionamento, e nella traduzione di tale livello in un numero, che per un computer significa un certo insieme di bit.

La bontà di un campionamento dipende essenzialmente da due parametri: la frequenza di campionamento e il numero di bit utilizzati. Per ottenere un segnale di eccezionale qualità i compact disc utilizzano una frequenza di circa 44000 Hz e 14 bit. È facile calcolare che una simile frequenza richiede una memoria davvero enorme, pertanto il computer si limita a valori decisamente inferiori, ottenendo comunque un risultato davvero ragguardevole.

Amiga è in grado di riprodurre suoni campionati fino a un massimo di 8 bit, mentre non ci sono limiti per la frequenza di campionamento.

Utilizzando il software Audiomaster sono possibili due scelte di frequenza: 9000 Hz e 18000 Hz. È evidente che la seconda frequenza fornisce risultati migliori, tuttavia richiede una memoria esattamente doppia.

Ricordiamo tra parentesi il teorema secondo cui per campionare un suono fino a una certa frequenza è necessario effettuare il campionamento alla frequenza doppia, cosa che nel nostro caso significa che si possono campionare suoni fino a 4500 Hz o 9000 Hz. Potrebbe sembrarvi un risultato di scarsa qualità se paragonato ai 20000 Hz tipici degli impianti Hi-Fi, ma vi assicuriamo che questo valore consente di ottenere ottimi risultati.

Anche il numero di bit influisce sulla qualità del suono, infatti un numero maggiore di bit riduce il rapporto segnale-rumore.

Terminiamo la digressione tecnica e torniamo a Easy Sound. Eravamo rimasti all'ascolto del segnale tramite l'altoparlante del monitor. Quando il nastro è giunto al punto desiderato, è sufficiente premere il tasto del mouse per iniziare il campionamento. La durata del campionamento deve essere fissata a priori scegliendo la memoria che si desidera destinare all'operazione.

Naturalmente un suono campionato occupa una quantità considerevole di memoria, pertanto in linea di massima non è possibile superare i dieci-venti secondi, che comunque sono sufficienti per qualunque applicazione. Se poi vi interessa campionare un suono per utilizzarlo come strumento per Sonix o per altri programmi musicali, vi renderete conto facilmente che pochissimi secondi di campionamento sono sufficienti, pertanto non esistono problemi di memoria.

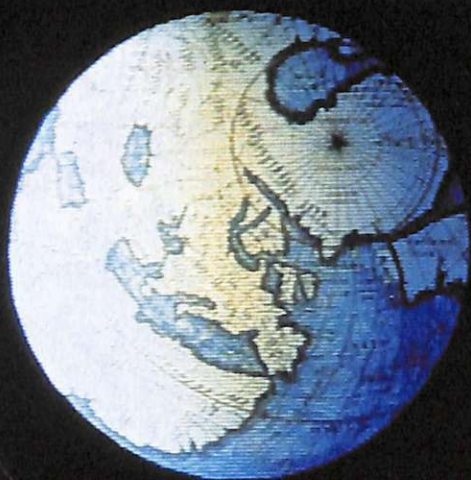
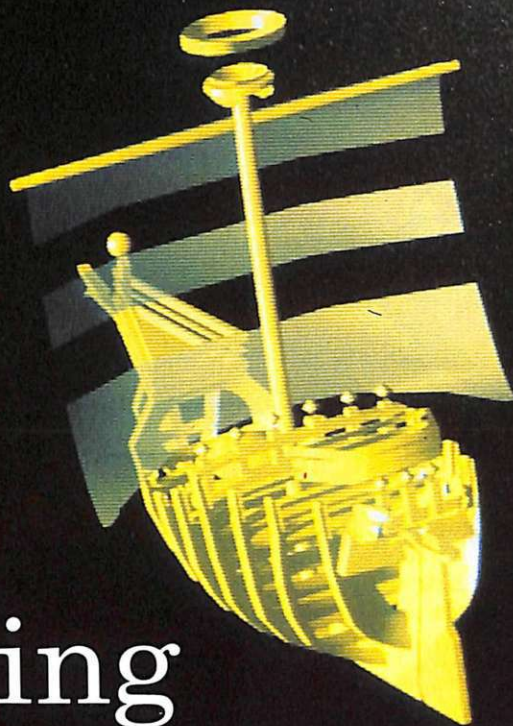
Naturalmente, dopo avere terminato il campionamento di un suono, è possibile effettuare qualsiasi operazione di editing: modificarne il volume, mixare due suoni, ottenere effetti di eco e tanti altri.

A questo punto però il compito di Easy Sound è terminato e il lavoro tocca al software utilizzato.

Questo accessorio è davvero un prodotto eccezionale: pur essendo uno dei campionatori più economici in commercio permette di ottenere gli ottimi campionamenti che si sentono talvolta nel software commerciale. Certo il software in dotazione non è un granché, il microfono è di poco valore e le istruzioni sono pessime, ma il campionatore vero e proprio è un ottimo prodotto che può soddisfare anche utenti molto esigenti.

Anche il prezzo è molto contenuto: 110.000 lire Iva compresa. Pertanto non possiamo che consigliarne l'acquisto a chiunque ami sentire il proprio computer esibirsi a viva voce.

Gianni Arioli

Easy Sound è disponibile presso la Niwa Hard & Soft, via B. Buozzi 94, Sesto San Giovanni (MI), tel. 02/2620312.

C64 in multitasking

In questa seconda puntata dedicata a Struct Basic i principali protagonisti sono le istruzioni di controllo e quelle per la gestione del multitasking. La programmazione concorrente d'ora in poi non sarà più un privilegio di Amiga ma sarà disponibile anche sul Commodore 64.

Nella puntata precedente è stato detto che Struct Basic supporta il multitasking, cioè mette in grado il C64 di eseguire più programmi Basic contemporaneamente. In questa puntata saranno descritte tutte le istruzioni che permettono di gestire questa utilissima capacità. Sono poche, molto semplici e potenti; quindi non dovrebbe essere difficile capirne il funzionamento e usarle subito nei vostri programmi, magari per creare effetti grafici irrealizzabili col Basic standard e con qualsiasi altra espansione.

Vedremo anche un'altra serie di istruzioni di controllo, altrettanto potenti come quelle viste nella puntata precedente, che vi aiuteranno a sviluppare più velocemente qualsiasi tipo di programma.

Esaminiamo ora la nuova serie di istruzioni che, come al solito, saranno suddivise secondo la loro funzione.

Comandi di supporto

• **Off**. Permette di disabilitare Struct Basic e quindi di riattivare l'interprete

standard. Dopo aver eseguito questa istruzione tutte le nuove istruzioni messe a disposizione da Struct Basic non saranno più utilizzabili da programma finché non caricherete e lancerete nuovamente l'espansione. L'effetto di questa istruzione è simile a quello prodotto dal reset, e quindi anche l'eventuale programma in memoria viene perso. Fate molta attenzione nell'uso di questa istruzione, perché può essere impiegata sia in modo diretto sia da programma e non richiede alcuna conferma malgrado sia molto pericolosa.

• **Clear.** Permette di cancellare una o più linee consecutive dello schermo. La sua sintassi è molto simile a quella dell'istruzione List del Basic standard, con la sola eccezione che al posto del trattino (-) va usata la virgola (,). Ecco qualche esempio d'uso:

- *Clear* (senza alcun parametro): cancella tutto lo schermo.
- *Clear 5*: cancella la sesta linea dello schermo. Infatti la numerazione delle linee di schermo parte da 0.
- *Clear 5,7*: cancella tre linee (la sesta, la settima e l'ottava).
- *Clear 4,*: cancella tutto lo schermo a partire dalla quinta linea. Le prime 4 linee dello schermo non vengono toccate.
- *Clear ,6*: cancella le prime sette linee dello schermo.

Questa istruzione può essere usata indifferentemente sia in modo diretto che in modo programma.

• **No stop**. Permette di disattivare il tasto Run/Stop e, se usata all'interno di un programma, lo rende inarrestabile. Questa istruzione, utilizzabile indifferentemente sia in modo diretto sia in modo programma, insieme a quella successiva permette di proteggere il vostro software dagli sguardi indesiderati (ovviamente il programma dovrà partire in autostart al termine del caricamento, altrimenti il listato potrà essere osservato tranquillamente prima di lanciare il programma). Non si tratta ovviamente di una protezione invincibile ma è pur sempre meglio che niente.

• **No restore**. Disabilita il tasto Restore. È molto utile se usata insieme all'istruzione precedente. No restore può essere usata sia in modo diretto sia in modo programma.

• **Admit stop**. Riabilita il tasto Run/stop. Questa istruzione serve per annullare l'effetto dell'istruzione No stop e può essere usata sia in modo diretto sia in modo programma.

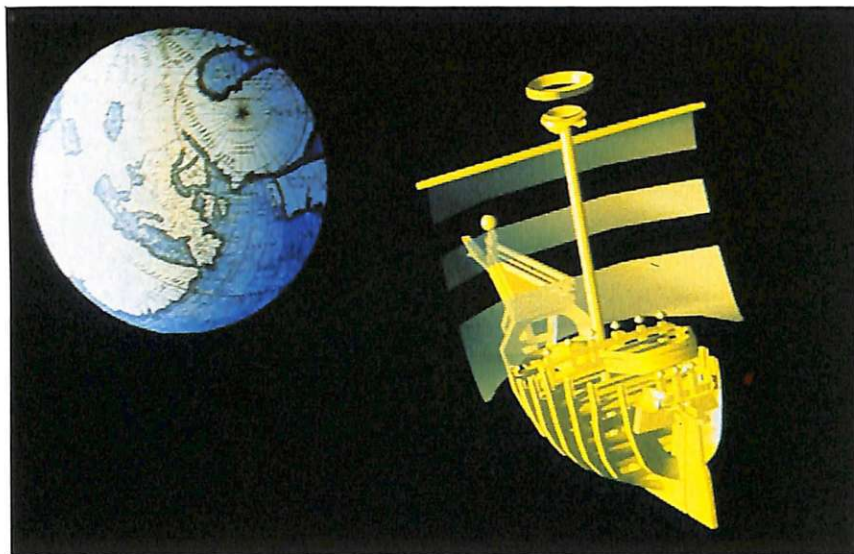
• **Admit restore**. Riabilita il tasto Restore. Questa istruzione ha ovviamente la funzione di cancellare l'effetto dell'istruzione No restore e può essere usata sia in modo diretto sia in modo programma.

• **Swap**. Si tratta di un'utilissima istruzione che ognuno di voi ha sicuramente sempre desiderato avere a disposizione. Permette di scambiare il contenuto di due variabili dello stesso tipo. La sua sintassi è: Swap var1, var2, dove var1 e var2 sono ovviamente le due variabili di cui si vuole scambiare il contenuto. L'istruzione non effettua nessun controllo sull'effettiva compatibilità di tipo delle variabili, quindi fate molta attenzione quando la usate. Ecco qualche esempio d'uso:

```
10 a=100
20 b=200
30 swap a,b
40 print a,b
50 a=a+1:b=b+1
60 print a, b
```

Dopo aver copiato questo listato, date il Run. Sul video verranno visualizzati nell'ordine i valori 200, 100, 201 e 101. L'istruzione Swap può essere usata sia in modo diretto sia in modo programma.

• **Sleep**. Pone in wait, cioè fa addormentare l'interprete Basic e quindi può essere usata per creare cicli di attesa temporizzati all'interno di un programma. La sintassi è: Sleep n, dove n rappresenta l'intervallo di tempo, in settantesimi di secondo (circa), durante il quale si vuole fare sospendere l'attività dell'interprete Basic.

Questo parametro può assumere un valore qualsiasi fra 0 e 65535. Sleep può essere usata anche in modo diretto e in questo caso ha un effetto simile a quello che si ottiene lanciando il seguente programma: 0 sleep n.

• **Clock**. Anche questa istruzione è utilissima, perché permette di manipolare facilmente l'orologio del sistema. La sintassi è: Clock "hhmmss". La stringa passata come parametro rappresenta l'ora su cui verrà regolato l'orologio. Questa istruzione può anche essere usata come funzione se la si usa senza parametri. Ecco un esempio:

```
10 clock "120000"
20 print"<home>" clock
30 goto 20
```

Copiando e lanciando questo programma si otterrà la visualizzazione dell'ora in tempo reale nell'angolo superiore sinistro dello schermo.

• **Up**. Permette di fissare il limite superiore dell'area disponibile per i programmi Basic. La sintassi è: Up n, dove n rappresenta l'indirizzo dell'ultima locazione di memoria utilizzabile da Basic. Questa istruzione si rivela particolarmente utile quando si devono realizzare programmi che fanno uso di molti sprite e caratteri programmabili, perché permette di riservare con estrema semplicità e precisione tutto lo spazio necessario per gli oggetti che si vogliono definire.

• **Start**. Permette di fissare l'inizio della zona di memoria disponibile per i programmi Basic. La sintassi di questa istruzione è identica a quella dell'istruzione precedente, con la sola differenza che in questo caso n rappresenta l'indirizzo della prima locazione utilizzabile da Basic.

Simulare il multitasking

Struct Basic mette a disposizione anche alcune istruzioni per simulare l'esecuzione contemporanea di più linee Basic.

Il multitasking sul Commodore 64, così come su ogni sistema dotato di un solo processore, può essere soltanto simulato e viene implementato con la tecnica del time slicing, cioè assegnando la CPU a ogni processo attivo per un determinato intervallo di tempo.

Vediamo subito un esempio pratico per capire meglio il procedimento:

```
10 print "<clr/home>"
20 irqjob 100,3
30 irqon
40 poke 1024+i,1
50 poke 55296+i,0
60 i=i+1
70 if i=1000 then i=0:t=0
80 goto 40
90 :
100 poke 2023-t,26
110 poke 56295-t,1
120 t=t+1
130 irqend
```

Copiate questo breve programma e lanciatelo. Lo schermo verrà riempito a partire dall'angolo superiore sinistro con la lettera a e dall'angolo inferiore destro con la lettera z.

Tuttavia, vedendo il listato, si dovrebbe credere che la parte di programma che visualizza la lettera z in realtà non può essere eseguita a causa dell'istruzione goto 40 alla linea 80. La soluzione a questo apparente dilemma sta nelle prime due linee del programma.

La prima linea contiene l'istruzione irqjob, che serve per definire l'inizio del blocco di linee (d'ora in poi useremo il termine task 1 per identificare questo blocco di linee) da eseguire contemporaneamente a quelle che normalmente devono essere interpretate (d'ora in poi useremo il termine task 0 per identificare questo blocco di linee), cioè le linee successive a quella che contiene l'istruzione irqjob. I due parametri che figurano in questa istruzione sono rispettivamente il numero che identifica la prima linea del task 1 e l'intervallo di tempo da dedicare al-

la sua esecuzione. Il primo parametro deve essere un numero di linea esistente, mentre il secondo deve essere un intero compreso fra 1 e 255. Il valore del secondo parametro è inversamente proporzionale al tempo dedicato all'esecuzione del task 1. Nell'esempio precedente il tempo dedicato al task 1 è quasi uguale a quello dedicato all'esecuzione del task 0. Tenete presente che non esiste la possibilità di creare un ambiente indipendente per ogni task, e quindi le variabili usate in un task sono visibili anche nell'altro task.

Ora vediamo l'elenco completo delle istruzioni disponibili per gestire il multitasking:

• **Irqjob**. Permette di definire la linea di inizio del task 1 e il tempo da dedicare alla sua esecuzione. La sintassi è: Irqjob ln, tm. Il primo parametro rappresenta la prima linea del task 1, mentre il secondo rappresenta l'intervallo di tempo da dedicare all'esecuzione del task.

• **Irqon**. Abilita l'esecuzione del task 1. Ovviamente questa istruzione dovrà essere usata solo in unione all'istruzione precedente.

• **Irqoff**. Disabilita il task 1.

• **Irqend**. Questa istruzione va sempre messa al termine del task 1 e ha la stessa funzione di un Return posto al termine di una subroutine. Quindi questa istruzione serve per consentire il ritorno al task 0.

• **Pullirq**. Questa istruzione può essere usata solo nel task 1; in pratica serve per trasformare il task 1 in una normale porzione di programma e quindi, al termine dell'intervallo di tempo dedicato al task 1, il controllo non tornerà più al task 0.

Istruzioni di controllo

Oltre alle istruzioni di controllo viste nella puntata precedente, Struct Basic ne mette a disposizione altre un po' più simili a quelle disponibili in Basic standard. Eccole:

• **Label**. Permette di etichettare una linea Basic con una stringa. Una linea Basic segnata con questa istruzione potrà essere chiamata mediante una particolare istruzione di salto, che vedremo in seguito, sia tramite il suo numero di linea sia per mezzo dell'etichetta a essa associata. La sintassi è: label "etichetta". In seguito vedremo un esempio di impiego di questa istruzione. Nei vostri programmi cercate di usare il più possibile le etichette, perché rendono il programma estremamente leggibile.

• **Jump**. È un'istruzione di salto simile al Goto del Basic standard. Si differenzia da quest'ultima istruzione perché deve essere seguita da una etichetta e non da un numero di linea. Ecco un esempio:

```
10 jump help
20 label continua
...
1000 label help
1100 print"Usa i tasti di movimento del
     cursore"
1110 print"per scegliere l'opzione"
1120 print"e return per confermare la
     scelta"
1130 jump continua
```

Da questo esempio si può vedere chiaramente che il guadagno in termini di chiarezza è notevole. Ovviamente tutto dipende dalla scelta dell'etichetta, che deve essere sempre significativa e non scelta a caso.

L'istruzione Jump può essere usata anche con un valore numerico, benché sia vivamente sconsigliabile.

Quindi nell'esempio precedente alla linea 10 si poteva mettere jump 1000 anziché jump help.

Struct Basic mette a disposizione molte altre istruzioni di controllo che dovrebbero limitare al massimo l'uso di istruzioni di salto. Tuttavia, se proprio non potete fare a meno di usare istruzioni di questo tipo, cercate di usare il più possibile l'istruzione Jump con una label piuttosto che Goto o Jump con un numero.

• **Call**. Questa istruzione equivale all'istruzione Gosub. L'unica differenza consiste nel fatto che Call può avere come argomento una etichetta. L'esempio precedente può essere migliorato utilizzando Call al posto di Jump e quindi aggiungendo un Return al termine della subroutine Help, quindi questa subroutine potrà essere usata in più parti del programma.

```
10 call help
...
...
1000  label help
1100  print"Usa i tasti di movimento del
      cursore"
1110  print"per scegliere l'opzione"
1120  print"e return conferma la scelta"
1130  return
```

Anche l'istruzione Call può essere usata con un numero di linea, oltre che con una etichetta. Tuttavia è molto più utile usare Call con una etichetta.

• **Pullsub**. Questa istruzione permette di rimuovere dallo stack il numero di linea salvato in seguito all'esecuzione dell'istruzione Call o Gosub. Il numero di linea salvato nello stack corrisponde alla linea da cui deve riprendere l'esecuzione quando verrà incontrata l'istruzione Return. È indispensabile usare questa istruzione quando si vuole uscire da una subroutine con un'istruzione di salto (Jump o Goto) e non si vuole che venga eseguita l'istruzione Return. In una situazione di questo tipo è assolutamente indispensabile usare l'istruzione Pullsub per aggiornare correttamente lo stack.

Istruzioni grafiche

• **Border**. Permette di cambiare il colore del bordo dello schermo. La sintassi di questa istruzione è: Border n, dove n è il codice del colore che si vuole assegnare al bordo dello schermo. Questa istruzione può essere usata sia in modo diretto sia in modo programma.

• **Paper**. Per di cambiare il colore del fondo dello schermo. La sintassi di Paper è identica a quella dell'istruzione precedente. Anche Paper può essere usata sia in modo diretto sia in modo programma.

• **Ink**. Permette di fissare il colore di linea, cioè il colore con cui verranno visualizzati i caratteri stampati con Print. La sintassi di questa istruzione è: Ink n, dove n è il codice del colore di visualizzazione desiderato.

Daniele Maggio

SE HAI PERSO UN NUMERO...
... HAI PERSO UN TESORO

Come fai se l'arretrato non ce l'hai?

Ti sei perso un numero, o addirittura più numeri, di RadioElettronica&Computer? In queste pagine ti viene offerta l'opportunità di rimetterti in pari. Di ogni arretrato troverai l'elenco dei progetti pubblicati quel mese. Affrettati a spedire la richiesta utilizzando il coupon pubblicato in queste pagine: riceverai subito a casa tua il numero o i numeri che ti interessano senza aggravio di spese postali.

1 - Gennaio 1988 - L. 10.000 - Prosegue il corso di scacchi. Agocom: il software che fa l'agopuntura. Banche dati: migliaia d'informazioni utili in tempo reale. Smart modem 21-22. Con Super Basic avrete 42 comandi per la gestione del drive e della grafica. Gioco: Paura nella torre. Basic Lightning: poligoni regolari, spezzate chiuse e aperte e i comandi per il movimentio degli sprite. Recensione: Street baseball. Gioco: Caccia all'obelisco. Fai da te: Comgun, bersaglio mobile. Magic Window: sempre più magica. Cruciverba: un programma creatore di schemi. Tips & Tricks: nuova rubrica dedicata ai lettori molto pratici con consigli di programmazione. Utility: una mini espansione per gli sprite.

2 - Febbraio 1988 - L. 10.000 - Comgun: software su cassetta per il tiro a segno presentato sullo scorso numero. Scacchi: tattiche antiche. Cruciverba: solutore elettronico. Circuiti con l'hard copy. Recensione: Combat school, simulatore di addestramento. Stampante novità della Citizen. Gioco: chi salverà la galassia? Archivio videocassette. Floracom: il computer pensa alle vostre piante. Gamebasic: per animare il gioco. Protector, la directory che rende illeggibile il contenuto dei vostri dischetti. Pet-Speed e Magic Window presenti sul vostro Commodore.

3 - Marzo 1988 - L. 10.000 - Scacchi: l'attacco di minoranza. Calcolo enigmatico. Banche dati: a Wall Street con The Source. Iconebase permette una facile archiviazione dei vostri dati. Recensione: è di turno il basket. Magic Window sempre più magica. Gioco: attenti agli animali robot. Gioco: forza uomo gatto. Fai da te: tutti atleti con Supergym. Disk Basic aggiunge ben 34 comandi al Basic per padroneggiare il drive. Il software professionale di Floracom. Didattica: ecco il primo appuntamento dedicato alla geometria analitica. Senza rimpianti il passaggio da Commodore a Amiga.

4 - Aprile 1988 - L. 10.000 - Basic Lightning: concludiamo l'analisi degli attributi. Wizard Key porta a 16 i tasti funzione. Eserciziario di trigonometria. Scacchi: un finale tra Re e pedone senza che venga giocato. Corsa spaziale in 3 dimensioni. Tanta RAM in una semplice cartridge. Recensione: Un gioco proprio come Platoon. Gioco: gli alieni invadono Urano. Segreteria telefonica intelligente. 4 routine per gli effetti speciali. Missione Gold Basic.

5 - Maggio/Giugno 1988 - L. 10.000 - Potete creare introduzioni grafiche degne dei più noti videogame. Phonextra: il software su cassetta della vostra segreteria telefonica. Interfaccia Amiga-like per il vostro C64. Music Basic: la più completa espansione di gestione del Sid. Gioco: 3 gorilla in città. Recensione: olimpiadi dagli effetti esilaranti. A tutta musica con il C64. L'espansione RAM migliora. Fai da te: la macchina della verità. Gioco: distruggete l'imperatore del sistema Delta. Nastroteca su disco. Scacchi: ultima puntata.

6 - Luglio/Agosto 1988 - L. 10.000 - Con Logic Val valutate complesse espressioni matematiche. Tecniche di caricamento di loader e turbo. Un disegnatore per i software sprite. BMX: una gara di biciclette. Gioco: Giana delle meraviglie. Digitalizzatore in real-time. Seconda puntata del corso di musica: scale musicali per pianoforte e la programmazione del SID. Un'utility che vi propone una griglia su cui disegnare. Drive basic, una super espansione. Ultima puntata del corso di scacchi. Recensione: Firefly, salvate la Terra dall'invasione nemica. Software su cassetta della macchina della verità. Una routine per scovare gli errori nei dischi. Prima parte degli esercizi sullo studio delle funzioni.

7 - Settembre 1988 - L. 10.000 - Basic test: 4 programmi per migliorare la conoscenza del basic. Dan Dare: un'avventura mozzafiato. Gli errori del computer. Fuga da Zart: il più irresistibile tra gli arcade. Le novità sul drive 1541 II. L'eserciziario di matematica: le funzioni trascendenti. Realizzare animazioni e giochi con Animation C.K. 1a puntata sullo standard MIDI. Grafik 2000: gestione avanzata dell'alta risoluzione. L'hardware del C64. Gestioni campionati. Recensione: Bedlam, uno spaziale con ben 16 schermi diversi.

8 - Ottobre 1988 - L. 10.000 - Action Replay e The Expert Trilogy: due cartucce dalle prestazioni eccezionali. Guerra in Vietnam: i dodici della squadriglia blu. Quattro caricatori per risolvere i problemi di duplicazione e trasferimento programmi da nastro a disco. Flying Shark II: una squadriglia di caccia nei cieli d'Africa. Ancora trucchi e scorciatoie per il tuo C64. Recensione: Il perfido Elvin vuole distruggere il mondo. Shoot'em up: un programma per realizzare videogames. Sprite & Graphic Basic: un'espansione per la gestione degli sprite e dell'alta risoluzione. L'architettura dei bus: il micropro-cessore. Gli "errori" del computer. Velox: un colloquio contemporaneo e diretto di ben sette persone con il computer. Come gestire al massimo le possibilità di Mastercard.

9 - Novembre 1988 - L. 10.000 - Più compatte le schermate del Koala. Il linguaggio dello standard Midi. Gioco: i misteri di Netherworld. Massima qualità con Sound 64. Roby Robot: l'hardware che pilota gli elettrodomestici. L'angolo di Amiga. Shoot'em up: gli sprite: Recensione: Target Renegade. Seconda puntata di SG Basic. Come gestire Velox.

10 - Dicembre 88/Gennaio 1989 - L.10.000 - Utility: 99 font per accessoriare i propri programmi. Midi: ecco come usare i messaggi. Fai da te: accendi col computer il tuo albero di Natale. L'intramontabile Mickey Mouse. SG Basic: la stesura dei programmi. Televideo nel C64. Shoot'em up: scenario ed effetti sonori. Computer Graphics con Amiga. Didattica: calcolo integrale di una funzione variabile. Arrow of death: l'avventura più bella.

1 - Febbraio 1989 - L. 10.000 - Avventura: i segreti di Arrow of Death. Tutti i segreti per fare word processing. Listati: l'utility per esporli e commentarli. Recensione: a scuola di guerra. Roby Robot, seconda puntata. Pagefox: la cartuccia per stampare. Fai da te: il joystick che obbedisce alla tua voce. Amiga: alla scoperta dell'hardware. Shoot'em up: come sistemare i protagonisti. Ultima puntata di SG Basic.

2 - Marzo 1989 - L. 10.000 - Easy Script: ultima puntata. L'hardware per la gestione dell'acquario. Sprite giganti Hi-Res. Text to text: per abbattere le barriere di incompatibilità fra Ascii, PetAscii e codici video. Il gioco dell'androide volante. Il software per far eseguire al joystick i comandi impartiti a voce. I linguaggi di Amiga.

3 - Aprile 1989 - L. 10.000 - Fai da te:Elettronica al servizio dello sport. Utility: campionamento di suoni col registratore. Gioco:Psi-droid lucido metallo. Espansione: Basic strutturato. Amiga:Il multitasking senza veli. Accessori:Programmatore di eprom.

Tagliando richiesta arretrati

Per ricevere a casa, senza aggravio di spese postali, l'arretrato o gli arretrati che ti interessano, compila e spedisci subito questo tagliando in busta chiusa a:

RadioElettronica&Computer
Gruppo Editoriale JCE
via Ferri 6 - 20092 Cinisello Balsamo (MI)

Sì! Inviatemi i seguenti numeri arretrati di RadioElettronica&Computer:

mese/mesi di ....

Cognome .... Nome ....

Via ....

Cap .... Città .... Prov ....

❑ Allego L. ....

❑ Allego ricevuta di versamento di L. .... sul conto corrente postale n. 351205 intestato a Gruppo Editoriale JCE - via Ferri 6 - 20092 Cinisello Balsamo (MI)

❑ Allego assegno di L. .... non trasferibile intestato a Gruppo Editoriale JCE

Data .... Firma ....

Quando il computer parla italiano

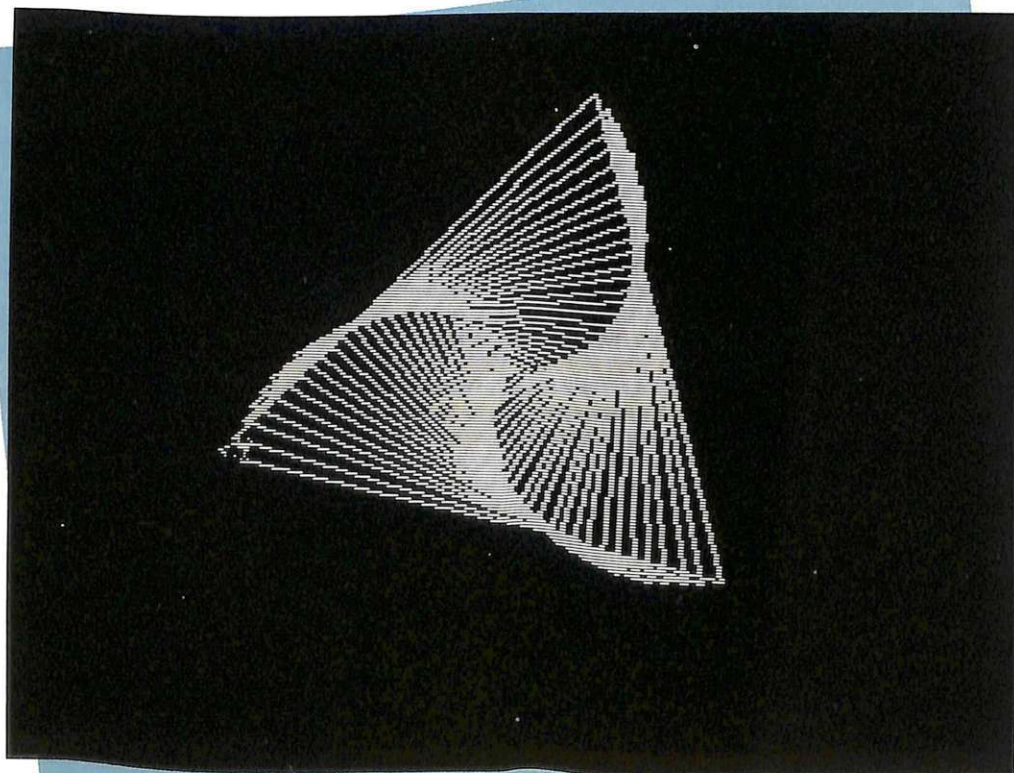
Inizia un corso dedicato a chi non ha mai programmato e desidera farlo con soddisfazione fin dai primi approcci. Il linguaggio di cui ci occuperemo è il Logo, famosissimo per il suo alto livello d'evoluzione, cioè per la sua vicinanza, in termini di struttura logica e formale, alla mente umana.

Il corso che ci proponiamo di sviluppare si rivolge a tutti gli utenti che non hanno molta domestichezza con il computer ma sentono la necessità di saperne di più e desiderano capire quei concetti generali che fanno della programmazione di un computer la più richiesta e ambita forma di pensiero logico del nostro tempo.

Per prima cosa caricate dalla cassetta il programma Logo (si trova, sul nastro, dopo il programma Dsave V2) digitando LOAD"LOGO", battendo il tasto Return e avviando il registratore premendo il tasto Play. Quando compare il messaggio di Ready battete RUN seguito dalla pressione del tasto Return e attendete che lo schermo diventi tutto azzurro con le scritte blu. Da questo momento in poi non dovete far altro che seguire le indicazioni che trovate in questo corso.

La nascita di Logo

Una domanda che molti insegnanti e informatici si sono posti in questi ultimi anni è la seguente: "Qual è il linguaggio di programmazione che più si adatta a permettere un facile e corretto approccio al mondo dei calcolatori e al pensiero algoritmico?".

Naturalmente vi sono state diverse risposte: a partire da coloro che professano un approccio all'informatica senza elaboratore (l'informatica con carta e matita), passando dai cultori del Basic, agli stilisti

Listato 1.

```
TANA PS <Return>
ASCOL 0 A 30 D 90 <Return>
ASCOL 1 A 35 D 90 <Return>
ASCOL 2 A 40 D 90 <Return>
ASCOL 3 A 45 D 90 <Return>
ASCOL 4 A 50 D 90 <Return>
ASCOL 5 A 55 D 90 <Return>
ASCOL 6 A 60 D 90 <Return>
ASCOL 7 A 65 D 90 <Return>
ASCOL 8 A 70 D 90 <Return>
ASCOL 9 A 75 D 90 <Return>
ASCOL 10 A 80 D 90 <Return>
ASCOL 11 A 85 D 90 <Return>
ASCOL 12 A 90 D 90 <Return>
ASCOL 13 A 95 D 90 <Return>
ASCOL 14 A 100 D 90 <Return>
ASCOL 15 A 105 D 90 <Return>
```

del Pascal, fino ad arrivare ai puristi del linguaggio macchina, tutti hanno trovato in ciascuno di questi linguaggi delle caratteristiche positive applicabili al campo educativo.

A questa folta schiera si è aggiunto ultimamente il gruppo dei profeti della tartaruga, ovvero i cultori del Logo. Il Logo è stato il primo linguaggio di larga diffusione ad essere stato studiato espressamente per un uso educativo ed è perciò naturale che la sua comparsa abbia destato grande interesse e altrettanto grandi polemiche. Interesse perché naturalmente tutti vogliono sapere quali sono i vantaggi del Logo rispetto agli altri linguaggi di programmazione e cosa significa la definizione di linguaggio didattico. Polemiche perché, essendo un linguaggio didattico, o pedagogico, si basa su determinate teorie pedagogiche e ogni teoria ha i propri estimatori e denigratori.

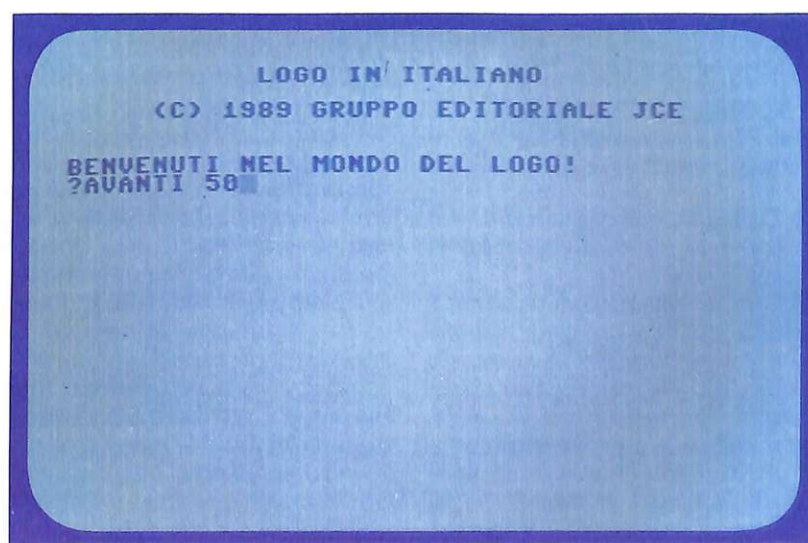
Vediamo di risalire alla nascita di Logo per capirne le radici e i propositi dei suoi realizzatori.

Il progetto Logo nasce agli inizi degli anni '70 nei laboratori del Massachusetts Institute of Technology, da un'idea di Seymour Papert, un matematico specializzato nell'applicazione dei computer nel campo dell'Intelligenza Artificiale e di applicazione dei computer nello studio dello sviluppo dell'intelligenza. Sotto il punto di vista pedagogico la sua impostazione deve molto alla collaborazione con l'epistemologo e pedagogista J. Piaget, avvenuta in Svizzera poco prima della sua morte. Per questa impostazione Logo si è posto immediatamente come momento di rottura con le precedenti applicazioni educative degli elaboratori, le macchine per insegnare e i programmi di istruzione programmata, che trovano le loro radici teoriche nella psicologia comportamentistica.

Dai suoi studi sullo sviluppo dell'intelligenza e dalla parallela esperienza con i computer, Papert comprende quale sia il potenziale educativo dell'utilizzo della logica di analisi insita nella programmazione dei computer. Programmare potrebbe per esempio aiutare a comprendere alcuni concetti matematici altrimenti difficili da capire: in generale il computer potrebbe essere considerato un meraviglioso laboratorio matematico, dove idee astratte possono essere sperimentate concretamente.

I computer dell'epoca sono però dei giganti poco amichevoli, hanno dimensioni considerevoli e parlano un liguaggio essenzialmente tecnico, riservato a una stretta cerchia di iniziati. Il primo passo verso l'utilizzo dei computer nell'educazione deve essere proprio il linguaggio. Allo stesso tempo la creazione di un nuovo linguaggio deve avere per forza di cose un punto di riferimento e, tra i linguaggi allora disponibili (Fortran, Cobol, Basic, Lisp e Assembler), Papert sceglie il Lisp.

Il Lisp è il linguaggio che viene utilizzato per risolvere problemi di intelligenza artificiale, quando cioè non occorre manipolare esclusivamente numeri, come avviene in campo scientifico, ma anche concetti e idee. Concetti e idee che si esprimono nella memoria di un computer sotto forma di liste di parole (da qui il nome LISP, cioè List Processor).

Il concetto base di Lisp viene ripreso, la sintassi del linguaggio viene semplificata in modo da essere accessibile a persone non specializzate e viene aggiunta la caratteristica che più ha reso Logo famoso, la grafica della tartaruga. Fino a quel momento la grafica con il computer permetteva di fare cose interessanti, ma richiedeva elevate capacità di calcolo matematico, in quanto il sistema utilizzato per rappresenta-

Questa è la schermata di apertura del programma Logo. Molti comandi Logo possono essere eseguiti semplicemente digitandoli sulla tastiera.

re figure sullo schermo di un computer altro non era che il solito sistema di riferimento delle coordinate cartesiane. Ciò rendeva difficile l'utilizzo di una risorsa così potente in campo educativo quale la grafica, ma il problema fu brillantemente risolto inventando un nuovo sistema di riferimento per la creazione di grafici, che non aveva bisogno di alcuna precedente conoscenza teorica, dato che per disegnare le figure si avvale di comandi semplici e naturali come Avanti, Indietro, Destra e Sinistra.

Così Logo si differenzia dai linguaggi tradizionali per tre motivi principali: la struttura di programmazione simile ai linguaggi per l'intelligenza artificiale, la grafica della tartaruga e la sintassi particolarmente semplice. Logo è rimasto per molti anni relegato nei laboratori dove è stato concepito, dato che solo ultimamente i piccoli computer sono diventati abbastanza potenti e accessibili economicamente da permetterne la diffusione su larga scala. Per fare il paragone con un altro linguaggio di programmazione molto diffuso basta notare che mediamente un interprete Basic occupa dagli 8 ai 16 Kb di memoria mentre per Logo ci vogliono almeno 28 Kb e notevoli capacità grafiche.

Ora che Logo è uscito dai laboratori ed è stato presentato come linguaggio potenzialmente ideale per le applicazioni educative, molte persone, anche autorevoli, hanno speso fiumi di inchiostro nel lodarlo e criticarlo.

Esercizi - Capitolo primo

Eccovi alcuni esercizi per il rafforzamento e l'approfondimento di quanto fatto nel capitolo primo. Sempre partendo dalla TANA:

1) Quanti passi deve fare la tartaruga per arrivare al bordo superiore dello schermo?

2) E per arrivare al bordo destro?

3) Segnate con un pennarello un punto sullo schermo, nella parte alta a destra. Provate a raggiungerlo con la tartaruga. Tenete nota dell'elenco dei comandi. (ricordate F1 per passare allo schermo Testo).

4) Segnate un punto in basso a sinistra e cercate di raggiungerlo. Confrontate gli elenchi dei comandi: in che cosa sono diversi? (nei comandi di spostamento o in quelli di direzione?).

5) Ripetete l'esercizio spostando il punto in basso a destra e poi in alto a sinistra. Questo vi serve per familiarizzare con la tastiera, lo schermo, e soprattutto con l'orientamento della tartaruga.

6) Segnate due punti sullo schermo, ai lati opposti, e partendo dalla TANA, girate attorno a un punto e raggiungete l'altro.

7) Partendo dalla TANA seguite un percorso a spirale fino a raggiungere il bordo.

A questo punto dovreste conoscere con sicurezza le dimensioni dello schermo, gli spostamenti angolari e la sintassi dei comandi.

8) Usando le primitive SU e GIU disegnate dei percorsi tratteggiati.

9) Disegnate le lettere dell'alfabeto.

10) Scrivete il vostro nome.

Logo in ambito didattico

Logo si differenzia dalla maggior parte delle altre applicazioni dell'elaboratore in campo didattico, esso si pone come strumento attivo per la sperimentazione e si basa fondamentalmente sul concetto che l'utilizzo dell'elaboratore come mezzo esplorativo della realtà e dell'analisi dei problemi sia di per sé utile allo sviluppo mentale. A causa di questa sua logica innovativa è anche molto più difficile programmarne l'inserimento in un tradizionale curriculum scolastico.

La sua introduzione nell'insegnamento di una materia qualsiasi presuppone una profonda analisi della disciplina stessa e un ripensamento sulle metodologie didattiche, dove invece i più tradizionali pacchetti di istruzione programmata non fanno altro che affiancarsi all'insegnamento tradizionale.

L'insegnante che voglia utilizzare Logo nell'insegnamento della sua disciplina difficilmente si troverà di fronte a qualcosa di pronto all'uso: egli dovrà sperimentare in prima persona le possibili applicazioni nel suo ambito di insegnamento. Questo naturalmente fino a quando un certo numero di persone non avrà fatto esperienza rendendo pubblici i risultati ottenuti.

A questo proposito vogliamo segnalare, nell'ambito dell'insegnamento della matematica e della geometria, il libro di H. Abelson e A. di Sessa, "Turtle Geometry, the Computer as a Medium for Exploring Mathematics" (MIT Press, Cambridge, 1981). Questo libro esplora le caratteristiche della Geometria della Tartaruga e vuole essere un'introduzione alla geometria in generale, che utilizza il computer come mezzo di esplorazione.

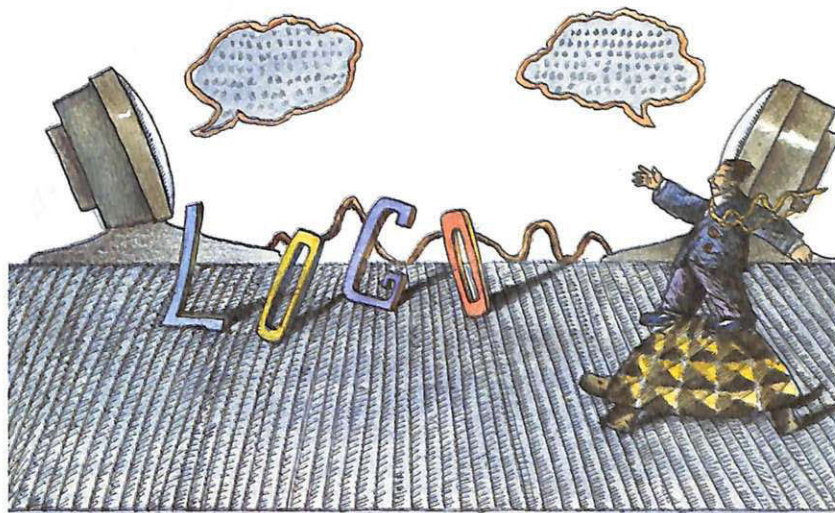
Gli autori sostengono che l'uso della sperimentazione diretta permette di assimilare concetti che usualmente vengono proposti in corsi più avanzati e stimola la nascita di una volontà individuale di esplorazione matematica.

Naturalmente, data la potenzialità di calcolo degli elaboratori, il punto di riferimento più immediato per l'educazione con l'elaboratore è nelle discipline matematiche.

Ultimamente però gli elaboratori sono stati messi in grado di manipolare anche informazioni non numeriche, cioè parole e liste di parole, e quindi il suo utilizzo è diventato possibile anche nel campo dell'analisi linguistica. Per esempio, insegnare al computer ad interpretare e capire frasi in linguaggio naturale può essere un modo molto efficace per capire e studiare la sintassi e la grammatica del linguaggio. Più in generale si può affermare che la risoluzione di problemi di intelligenza artificiale può aiutare a capire la logica formale e ad assimilare una metodologia di analisi dei problemi.

A livello di scuola elementare, Logo si presenta come un linguaggio ideale, data la sua semplicità sintattica, la struttura discorsiva delle sue istruzioni e l'immediatezza dei risultati ottenibili. Il fatto stesso di avere Logo in versione italiana è un punto a suo favore, in quanto abbatte la barriera linguistica, che a questi livelli rappresenta un ostacolo notevole. Per i bambini in questa fascia di età l'affrontare problemi come la costruzione di semplici figure con la grafica della tartaruga è un problema risolvibile, che introduce il bambino a una logica di pianificazione nella risoluzione dei problemi, oltre ad aiutarlo nella formazione di abilità quali la scomposizione di un oggetto nelle sue forme di base e la relazione tra le forme nello spazio, per citare solamente i più evidenti.

Questi esempi non vogliono naturalmente esaurire i possibili utilizzi didattici di Logo ma forniscono solamente una traccia e uno stimolo di ulteriore approfondimento e di analisi.

Passiamo adesso a un esame approfondito, diviso per capitoli, dell'utilizzo pratico di questo linguaggio.

Capitolo Primo

Scrivere con la tastiera

Per comunicare con il computer dovete usare la tastiera: non è difficile, basta fare attenzione alla posizione delle varie lettere sui tasti, e imparare alcuni trucchi per risparmiare tempo o per correggere se è necessario.

Ricordatevi che per il computer la lettera O non può essere usata per scrivere il numero 0 e lo stesso vale per la lettera I che non può essere usata per scrivere il numero 1.

Il Logo prevede due modi di scrittura: diretto e editor. Nel modo Diretto tutto quello che si scrive viene interpretato ed eseguito non appena si batte Return. Nel modo Editor, come approfondiremo nei prossimi capitoli, quello che si scrive va a formare una procedura, cioè una specie di programma, che non è eseguito direttamente ma viene depositato nella memoria del computer.

Nel modo Diretto si può scegliere di lavorare con la grafica (la famosa Tartaruga) o con le parole: in questi primi capitoli proveremo a lavorare con la tartaruga, perciò la prima cosa da fare è passare nel modo Grafica digitando il comando:

DISEGNA

seguito dalla pressione del tasto Return.

Che cosa succede a questo punto? Lo schermo si è diviso in due parti e in quella in basso è apparso il segno con il cursore lampeggiante. Nella parte in alto, nel centro, è apparso un triangolino che simboleggia la Tartaruga. Facciamole subito fare qualcosa, provate per esempio a digitare:

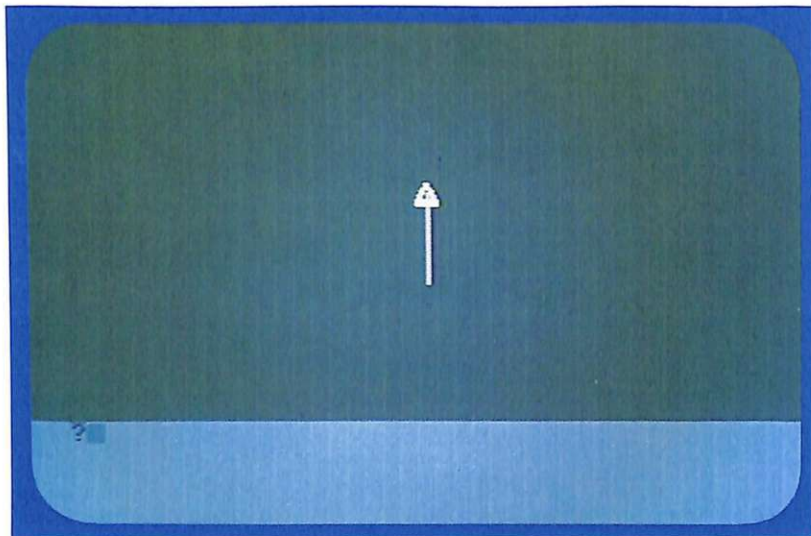
AVANTI 50

sempre seguito dalla pressione di Return. Controllate sullo schermo: la tartaruga si è mossa verso l'alto di qualche centimetro e ha lasciato come traccia una riga sottile. Se non è successo niente ed invece è apparsa una scritta come questa:

LA PROCEDURA AVANTI 50 NON ESISTE

è segno che non avete comunicato bene le vostre intenzioni alla tartaruga, ad esempio potreste esservi dimenticati lo spazio tra AVANTI e il numero 50, oppure aver compiuto qualche altro errore di battitura. Niente paura, ripetete l'esercizio finché vi riesce.

Provate adesso con quest'altro esercizio:

DESTRA 90 <Return>

Premendo il tasto Return dopo aver digitato un comando, questo viene immediatamente eseguito.

La tartaruga si è girata verso destra senza spostarsi. Con:

SINISTRA 90 <Return>

la tartaruga si gira verso sinistra, sempre senza spostarsi; mentre se digitate:

INDIETRO 50 <Return>

la tartaruga ritorna indietro, al punto di partenza e la sua traccia è rimasta sullo schermo.

Da questo primo esperimento potete ricavare alcune utili informazioni.

Per far muovere la tartaruga bisogna battere sulla tastiera alcuni semplici comandi: una indicazione di spostamento o di cambio di direzione e un numero che indica la quantità di spostamento o di cambio di direzione.

Queste informazioni, o comandi, o istruzioni, o parole, o come volete chiamarle voi, devono essere scritte con esattezza e separate tra loro da uno spazio.

Inoltre per poter essere eseguite devono essere comunicate al computer mediante il tasto Return, altrimenti non succede proprio niente!

Un'altra osservazione: mentre nei cambiamenti di direzione la tartaruga si limita a girare su se stessa, negli spostamenti produce una traccia che rimane sullo schermo.

Infine avrete osservato che se sbagliate qualcosa compare subito un messaggio d'errore. Capire i messaggi d'errore è un esercizio di grandissima utilità.

I primi vocaboli

Le parole che avete usato finora, assieme a quelle che vedremo fra poco, fanno parte del vocabolario della tartaruga e si chiamano Primitive per distinguerle da altre parole che non sono conosciute dalla tartaruga ma vengono create da chi usa il Logo.

Le primitive sono state inserite dagli autori del linguaggio per renderlo immediatamente utilizzabile anche dai non esperti. I nomi delle primitive risentono del paese di origine del programma quindi, a seconda della versione di Logo in nostro possesso, avremo primitive in inglese, tedesco, italiano, eccetera.

Come avete potuto osservare le primitive di Logo, nella versione italiana naturalmente, hanno un significato immediatamente comprensibile anche se questo può comportare altri problemi.

Per pulire lo schermo ad esempio si usa:

PULISCI SCHERMO <Return>

È una parola piuttosto lunga e dopo un po' diventa noioso usarla per intero.

Per fortuna il Logo accetta anche delle abbreviazioni e quindi PULISCI SCHERMO diventa la più comoda:

PS <Return>

Anche le primitive che avete usato finora si possono abbreviare:

- AVANTI diventa A
- INDIETRO diventa I
- DESTRA diventa D
- SINISTRA diventa S

Invece la primitiva:

TANA <Return>

che serve a riportare la tartaruga al centro dello schermo e rivolta verso l'alto, non si può abbreviare. La coppia di primitive:

TANA PS <Return>

è molto comoda quando si vuole iniziare un disegno con la tartaruga al centro e con lo schermo pulito. Può essere sostituita dalla primitiva:

DISEGNA <Return>

che svolge le stesse funzioni.

In origine la tartaruga era una vera e propria tartaruga meccanica che si muoveva con le ruote su un foglio di carta e poteva abbassare una penna per lasciare una traccia.

Di queste antiche funzioni sono rimasti i nomi di due primitive molto comode:

SU <Return>
GIU <Return>

che servono rispettivamente per far muovere la tartaruga senza

che lasci traccia e viceversa. Una altra primitiva interessante è:

ASCOL ...

che serve a determinare il colore della traccia lasciata dalla tartaruga. I puntini stanno ad indicare che ASCOL deve essere sempre seguita da un numero (da 0 a 15) che indica il colore scelto. Provate un po' a scrivere quello che vedete nel **listato 1**.

Se avete un monitor a colori noterete che alcuni colori sono meno belli degli altri: tenetene conto per quando vorrete fare dei disegni complessi e variopinti.

Ripetere senza fatica

Nell'esercizio precedente avete dovuto ripetere per molte volte una sequenza di comandi quasi uguali. C'è un modo per fare meno fatica, provate cioè a battere:

TANA PS <Return>
A 50 D 90 <Return>

Adesso premete il tasto con scritto CRSR U e la freccia in su, tenendo contemporaneamente premuto il tasto Shift (d'ora in poi lo chiameremo semplicemente CRSR-U). L'ultima riga è stata ripetuta nella parte in basso. Premete ancora due volte CRSR-U seguito Return: eccovi un bel quadrato!

A questo punto potrebbe avervi preso la curiosità di sapere dove vanno a finire le righe di istruzioni che scompaiono verso l'alto ogni volta che ne aggiungete una in fondo allo schermo. Provate a fare così:

• Premete il tasto F1. Lo schermo per la grafica è scomparso e al suo posto c'è solo testo (quello che avete scritto finora).
• Se premete F3 potete tornare allo schermo grafico e rivedere il vostro quadrato, con le comode righe per l'inserimento del testo.
• Se invece premete F5 avrete tutto lo schermo a disposizione per i disegni.

Purtroppo le righe sono troppe e, oltrepassando la parte alta dello schermo, andranno perdute e non c'è più modo di recuperarle. Non solo, vi sarete accorti che non si possono più riutilizzare i comandi già eseguiti prima dell'ultima riga. È una scomodità, ma non per molto. Promesso!

Come correggere

Nel mondo del Logo sbagliare è quasi la norma, l'importante è sapersi correggere.

Nel modo Testo, quando cioè scrivete qualsiasi cosa con la tastiera, la correzione si effettua utilizzando i tasti CRSR per muoversi lungo la riga, e il tasto DEL per cancellare a sinistra del cursore. Quando volete aggiungere una lettera o una parola, all'interno di una riga, portatevi con i tasti CRSR sul punto di inserimento e scrivete: le lettere si inseriranno automaticamente e sposteranno a destra quelle preesistenti. Provate a scrivere:

AVANTI 50

Correggete il 50 sostituendolo con 70: portate il cursore con i tasti Shift e CRSR sopra il 5 e scrivete 70. Come potete osservare il 70 si è inserito prima del 50, e adesso sullo schermo avete 7050 con il cursore sopra al 5. Spostatelo a destra dopo l'ultimo 0 e con il tasto DEL premuto due volte cancellate il 50. Sembra un po' complicato, ma è solo un'impressione: dopo un paio di volte vi sembrerà semplicissimo.

Correggere una parte del disegno invece è più complesso; spesso conviene pulire lo schermo con il comando:

PS <Return>

e riscrivere la sequenza corretta dei comandi. Ma se proprio volete cancellare l'ultima riga tracciata potete usare il comando:

ASCOL -1

e far tornare la tartaruga all'inizio della riga con il comando:

INDIETRO ... (scrivendo al posto dei puntini il numero di passi necessari)

Ricordatevi poi di riassegnare alla traccia il suo colore originario sempre con il comando:

ASCOL...(numero di colore)

Capitolo Secondo

Le procedure

Nel capitolo precedente avete scoperto che la tartaruga esegue le istruzioni che le vengono comunicate attraverso la tastiera. Le parole che avete usato sono quelle che la tartaruga è in grado di capire perché ne conosce già il significato. Non sono molte, è vero, pe-

Tavola 1.

```
PER QUADRATO <Return>
RIPETI 4 [A 50 D 90] <Return>
FINE <Return>
<Run/Stop>

PER SOLE <Return>
RIPETI 50 [A 100 D 100] <Return>
FINE <Return>
<Run/Stop>

PER CORONA <Return>
RIPETI 12 [STELLA RIPETI 30 [SU A 1 D 1 GIU]] <Return>
<Run/Stop>
```

rò è possibile inventarne di nuove e farle imparare alla tartaruga.

Prendiamo ad esempio questa serie di comandi:

TANA PS <Return>
A 50 D 144 <Return>
A 50 D 144 <Return>
A 50 D 144 <Return>
A 50 D 144 <Return>
A 50 D 144 <Return>

Il risultato è una bella stella! Peccato che per farne un'altra si debba scrivere di nuovo tutta quella serie di istruzioni.

Vediamo se si può fare qualcosa: se ci fate caso per ottenere la stella avete dovuto ripetere per 5 volte A 50 D 144 <Return>.

Perché non farlo fare alla tartaruga? Scrivete quindi:

TANA PS <Return>
RIPETI 50 [A 50 D 144] <Return>

Visto? Attenzione però alla regola: l'istruzione Ripeti deve essere sempre seguita da un numero (che indica quante volte si deve ripetere), e da uno o più comandi racchiusi tra parentesi quadre che rappresentano quello che deve essere eseguito. E adesso provate questo:

TANA PS <Return>
RIPETI 50 [A 100 D 100] <Return>

E quest'altro:

TANA PS RIPETI 4 [A 70 D 90] <Return>

Se lo desiderate potete scrivere i comandi tutti di seguito e senza andare a capo, purché vi ricordiate di separarli con uno spazio. E adesso cosa succede? La tartaruga ha forse imparato qualcosa? Un po' di pazienza! Per farle impare qualcosa dovete prima dirle di imparare. Volete insegnarle a fare una stella? Scrivete:

PER FARE.LA.STELLA <Return>

Lo schermo è diventato scuro e la tartaruga è scomparsa. In alto è apparsa la scritta:

PER FARE.LA STELLA

Adesso scrivete:

RIPETI 5 [A 50 D 144] <Return>

non è successo niente, eccetto il fatto che il cursore è andato a capo una riga sotto. Ora scrivete:

FINE <Return>

A questo punto avete composto la sequenza di istruzioni per ottenere la stella.

Ora potete dire alla tartaruga di impararla a memoria premendo il tasto Run/Stop. Lo schermo è diventato chiaro ed è comparsa la scritta:

FARE.LA.STELLA È DEFINITA

Questo significa che avete creato una nuova parola "fare la stella" e che questa è stata incorporata nel vocabolario della tartaruga. Provate adesso a scrivere:

FARE.LA.STELLA <Return>

Ed ecco apparire la vostra stella: da questo momento ogni volta che vorrete una stella potrete chiederla alla tartaruga semplicemente dicendoglielo.

Dopo un po' di volte forse vi sarete accorti che scrivere ogni volta FARE.LA.STELLA è piuttosto noioso: perché non insegnate alla tartaruga una nuova parola che vi risparmi la fatica? Provate:

PER STELLA <Return>
FARE.LA.STELLA <Return>
FINE <Return>

e ora premete Run/Stop. La procedura Fare.la.stella è stata conglobata nella procedura STELLA, così è sufficiente scrivere:

STELLA <Return>

perché la tartaruga, eseguendola, esegua anche FARE.LA.STEL-

Esercizi - Capitolo secondo

1) Scrivete una procedura per ottenere un trapezio.

2) Scrivete una procedura per disegnare l'iniziale del vostro nome.

3) Scrivetene un'altra per l'iniziale del vostro cognome.

4) Scrivetene una terza che faccia uso delle ultime due, e che scriva le vostre iniziali al centro dello schermo.

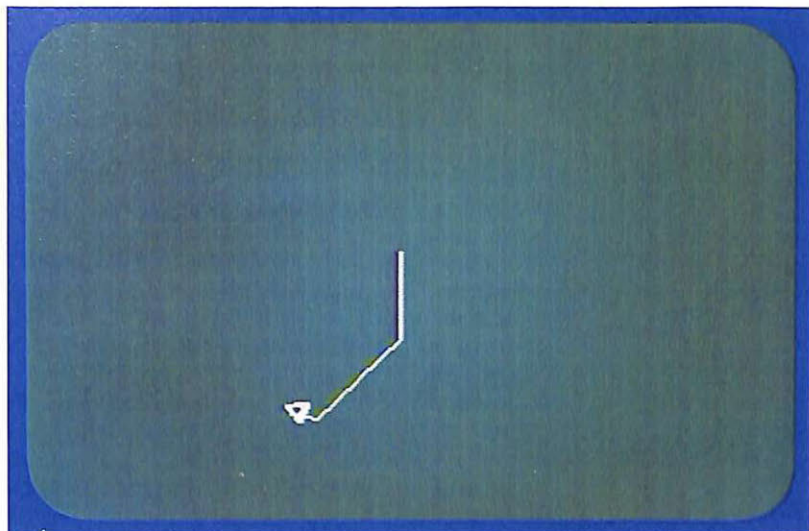
5) Partendo da questo comando: RIPETI 45 [A 1 D 1] <Return>, cercate di esplorare il mondo degli archi.

6) Provate a disegnare un fiore con queste procedure:
PETALO
SEMIPETALO
COROLLA
GAMBO.

7) Partendo da un arco arrivate al cerchio.

8) Provate a disegnare cerchi di dimensioni diverse.

9) Progettate una procedura che disegni un'onda.

10) Progettate una procedura che disegni una riga d'onde e poi un mare.

Lo schermo può essere solo testo, testo e modo disegno oppure solo disegno come in questo caso.

LA. Provatene qualche altra, aiutandovi con la **tavola 1**. L'ultima procedura della tavola 1 produce un disegno confuso perché le stelle sono troppo grandi, perciò bisogna intervenire all'interno della procedura per modificarla.

Modificare

Potete correggere o modificare una procedura senza doverla riscrivere da capo, chiamandola nell'Editor, cioè scrivendo: E FARE.LA.STELLA <Return>. Sullo schermo dell'Editor è ricomparsa la procedura in questione (perché questa e non Stella oppure Corona? Perché la stella vera a propria viene disegnata qua dentro, in FARE.LA.STELLA).

Spostate il cursore sulla parte che volete cambiare: in questo caso il 50 di RIPETI 5 [A 50 D 144], e scrivete 20.

Osservate che il 20 si inserisce spostando a destra il 50, portate il cursore sopra lo spazio che esegue il 50 e con il tasto DEL cancellate prima lo 0 e poi il 5. Adesso dovreste avere sullo schermo:

```
PER FARE.LA.STELLA
RIPETI 5 [A 20 D 144]
FINE
```

Se è così, confermate l'edizione della nuova procedura premendo il tasto Run/Stop.

Il mondo dell'Editor

Il mondo dell'Editor è un mondo un po' diverso da quello della Grafica o da quello del Testo: tutto quello che voi scrivete non viene eseguito immediatamente, e neppure vi vengono segnalati gli errori (se per caso ne fate), ma tutto rimane sullo schermo e basta.

Se le righe scritte sono più di quelle disponibili sullo schermo scorrono verso l'alto, ma non scompaiono come nel modo Testo perché potete sempre recuperarle usando i tasti CRSR per scorrere su e giù.

Tutto sembra inattivo finché non premete il fatidico tasto Run/Stop. In quel momento tutto quanto è nell'Editor viene "imparato" dal Logo e rimane nella memoria pronto per essere eseguito ogni volta che è necessario. L'Editor quindi è un potente strumento per la creazione e la modifica delle procedure e per sfruttare al massimo il potenziale di Logo.

A volte però potreste pentirvi di qualche modifica fatta nell'Editor e voler tornare indietro senza che succeda niente: premete allora i tasti CTRL e G e tutto tornerà com'era prima che chiamaste l'Editor. La coppia CTRL-G è di grande utilità anche in altre circostanze, ad esempio per fermare l'esecuzione di una procedura.

Conservare il lavoro fatto

Creare procedure è il modo normale di lavorare in Logo, anche se per rendere più sicuro e comodo il lavoro avete bisogno di conservare le vostre creazioni in un posto più affidabile e permanente di quanto sia la memoria del computer. Infatti, come ben saprete, se si toglie l'alimentazione al computer o lo si spegne, la memoria di lavoro (la famosa RAM) viene cancellata, perdendo tutte le informazioni che conteneva.

Per evitare questa eventualità è bene, ogni qualche ora di lavoro, mettere da parte il lavoro fatto. Per questo dovete procurarvi un dischetto vuoto e formattato, inserirlo nel drive al posto del disco Logo e procedere al salvataggio delle procedure con il comando:

CONSERVA "... nome scelto da voi <Return>

In questo modo tutte le procedure presenti in memoria vengono registrate sul disco. Per recuperarle e riutilizzarle scrivete:

RECUPERA "...nome <Return>

Per vedere l'elenco dei programmi presenti sul disco è sufficiente digitare:

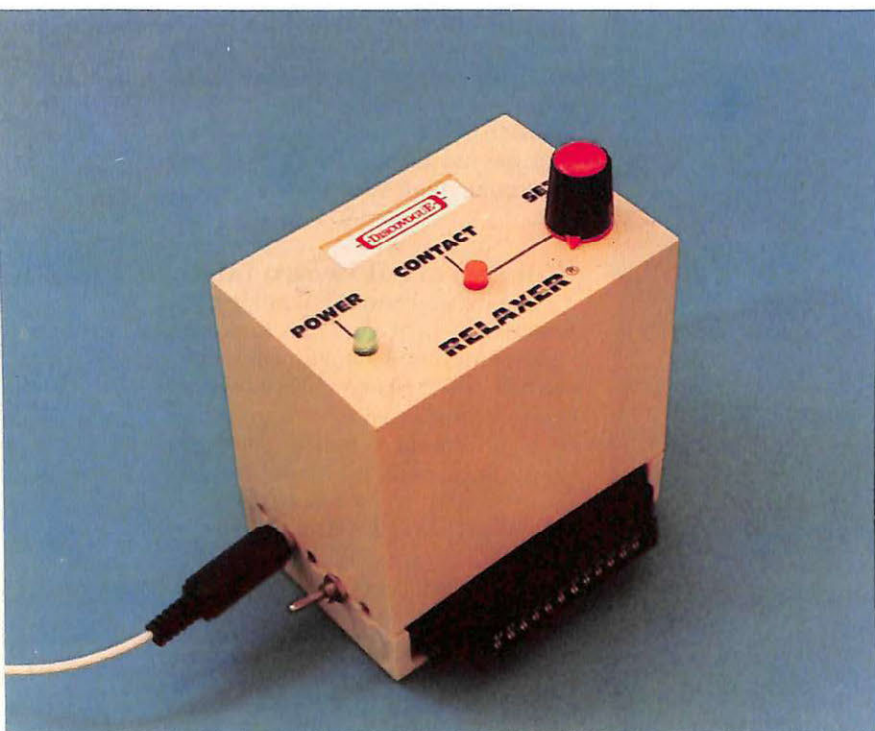
CATALOGO <Return>

Se invece una procedura non vi piace più o vi crea confusione, cancellatela dalla memoria con:

CANCELLA ... nome della procedura <Return>

abbreviato in: CA ...nome della procedura <Return>

Fernando Zanini

(continua)

Da oggi la vostra camera sarà un'oasi in cui rifugiarvi lontano dallo stress della vita moderna. Sulla base di considerazioni scientifiche quali la relazione fra il grado di umidità delle dita e lo stato di agitazione, abbiamo realizzato questo hardware, la cui trattazione, incluso il software, si esaurisce su questo numero.

Un'oasi nel deserto Caos

Relaxer è un dispositivo elettronico che, se applicato alla porta utente I/O multicanale del personal computer Commodore 64 (o del C128 perfettamente compatibile), con l'ausilio di appropriato software è in grado di rilevare, tramite uno speciale sensore a due terminali da applicare alle dita di una mano, le condizioni di rilassamento psico-fisico di una persona sottoposta a misurazione, sfruttando a tal scopo il continuo mutare delle condizioni di umidità superficiale cutanea (e dunque della conducibilità elettrica), tanto più elevata quanto maggiore è lo stato di disagio derivante da emozioni, ansia, eccitazione, affanno, affaticamenti.

Un segnale a frequenza variabile viene generato e trasmesso (tramite la linea di input PB0) al computer, che provvede a fornire un preciso, istantaneo e veritiero responso in base alle elaborazioni effettuate sul segnale stesso e ai confronti parametrici con valori di riferimento.

La frequenza di emissione del segnale impulsivo dipende dalla regolazione del potenziometro Set (da tarare prima di ogni seduta): le variazioni relative sono invece quelle dovute al modificarsi dell'umore del soggetto esaminato.

Particolarmente vantaggiosa risulta essere l'accoppiata hardware-software, che ha permesso la totale eliminazione della sezione circuitale relativa al controllo delle funzioni (display numerici, selettori, pulsanti), assimilandola direttamente al programma che gira su computer, con visualizzazione immediata su video delle procedure svolte o in corso: il tutto con un lay-out infinitamente migliore di quello ottenibile con i tradizionali pannelli elettronici di comando e controllo.

Una sicura interfaccia ottica incorporata, realizzata con un fotoaccoppiatore integrato, garantisce un totale isolamento tra apparecchio Relaxer e computer.

Anche per quanto riguarda l'alimentazione si può parlare di totale sicurezza e di piena autonomia: il circuito funziona con una tensione di 9 Volt, fornita da una pila esterna all'occorrenza sostituibile.

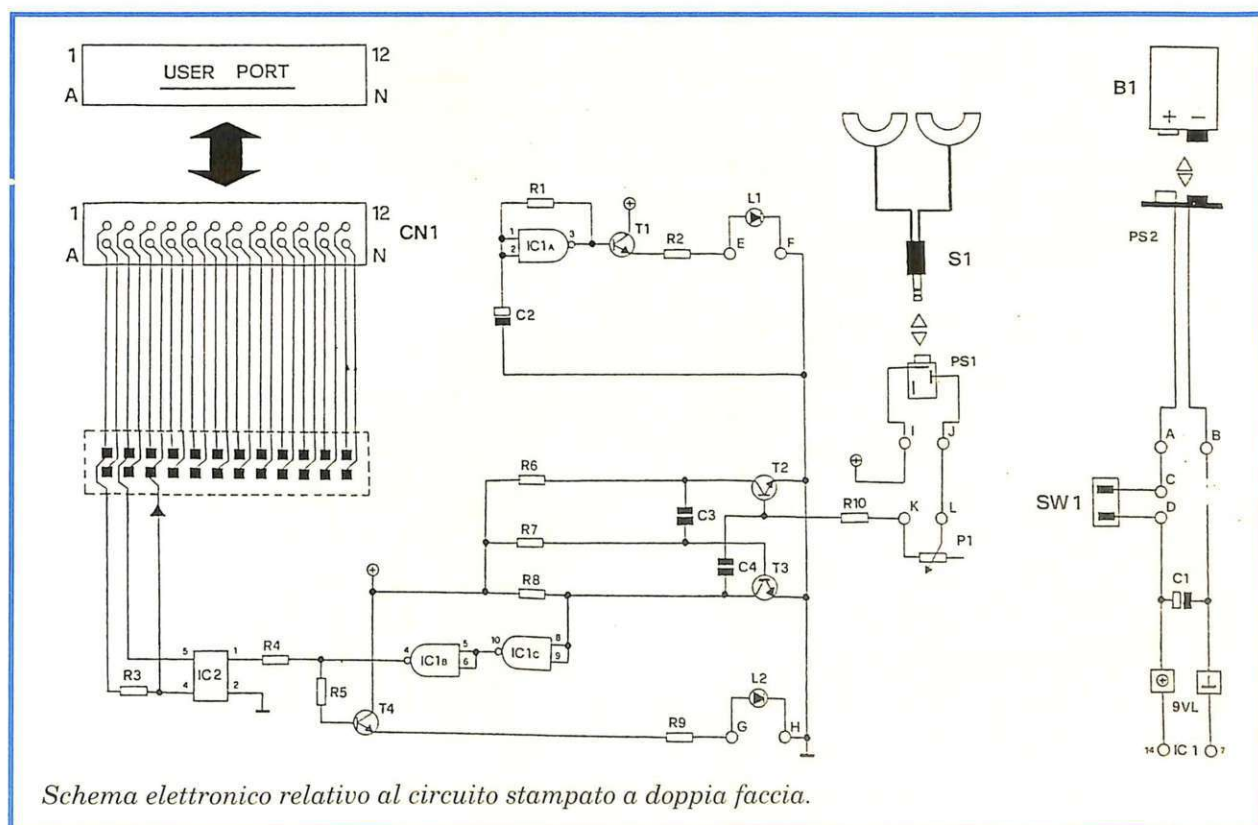
Il monitor di segnalazione ottica è composto da un led verde lampeggiante (Power), che segnala con continuità all'utente la presenza della tensione di alimentazione e il corretto funzionamento di tutto il dispositivo, e da un led rosso (Contact) che indica, lampeggiando più o meno velocemente, l'avvenuto contatto tra i due terminali del sensore e le dita della mano del soggetto analizzato.

Analisi di funzionamento

Il circuito elettronico di Relaxer comprende, oltre a un semplice stadio alimentatore (9 VL c.c. forniti da una pila e filtrati dal condensatore elettrolitico C1), anche un settore logico generatore di impulsi variabili ad alta frequenza costruito sui transistor T2 e T3: la variabilità è data dal potenziometro P1 Set che, opportunamente regolato, determina un valore centrale di riferimento.

Un sistema di conversione tensione-frequenza rende poi possibile l'aumento o la diminuzione degli impulsi (rispetto al valore centrale) in relazione all'input fornito dal sensore S1, dotato di due anelli metallici cortocircuitabili se messi a contatto con la pelle umana (per esempio con due dita di una mano). La tensione trasmessa tramite R10 risulta dunque essere tanto maggiore quanto più elevata è l'umidità superficiale cutanea. Ne consegue che sul collettore di T3, e quindi agli ingressi e sull'uscita del rettificatore NAND formato da IC1c e IC-1b (pin 8-9 e 4 di IC1) è presente il segnale impulsivo direttamente proporzionale allo stato emozionale del soggetto analizzato, segnale trasmesso all'ingresso del fotoaccoppiatore d'interfaccia IC2 (pin 1) e alla base del transistor T4 che pilota, tramite R9, il led rosso L2 Contact di monitoraggio. Alla linea di connessione sono riportati i 24 pin che consentono al computer collegato di comunicare (in input) con il circuito di Relaxer. Il collegamento con la linea dati della porta utente è realizzato attraverso i tre terminali 1, 2 e C, ovvero 0 Volt (1), 5 Volt positivi (2) e PB0 (C): si tratta di una connessione soltanto ottica, in quanto è presente un'interfaccia realizzata con il fotoaccoppiatore IC2.

Il totale isolamento tra computer e dispositivo Relaxer è garanzia di sicurezza operativa, ed evita il sorgere di disturbi e interferenze, sempre possibili quando si eseguono trasferimenti di segnali da un apparecchio a un altro. Durante il funzionamento attivo del software di gestione, non appena vengono ricevuti i dati impulsivi, il computer provvede a elaborare e a visualizzare un adeguato responso. Il led verde L1

Schema elettronico relativo al circuito stampato a doppia faccia.

Power, pilotato dall'oscillatore IC1a tramite T1 ed R2, lampeggiando alla frequenza determinata da R1 e C2, segnala all'utente la regolare presenza della tensione di alimentazione (9 Volt c.c.). L'interruttore SW1, collegato ai punti C e D del circuito, consente di accendere o spegnere comodamente l'apparecchio Relaxer senza dover applicare e togliere ogni volta la pila B1: questa dev'essere sempre carica almeno all'80 per cento, per garantire un buon funzionamento.

Assemblaggio circuitale e installazione

Il montaggio va iniziato solo avendo a disposizione tutto il materiale originale indicato nell'elenco componenti, e in particolare il circuito stampato a doppia faccia (cod. 171.66): l'osservanza di questa prima importantissima precauzione consente di portare a termine il lavoro senza che poi abbiano a verificarsi fastidiosi inconvenienti per mancato funzionamento dell'apparecchio autocostruito, quasi sempre causati dall'utilizzo di componenti diversi da quelli indicati, più che da veri e propri errori di montaggio.

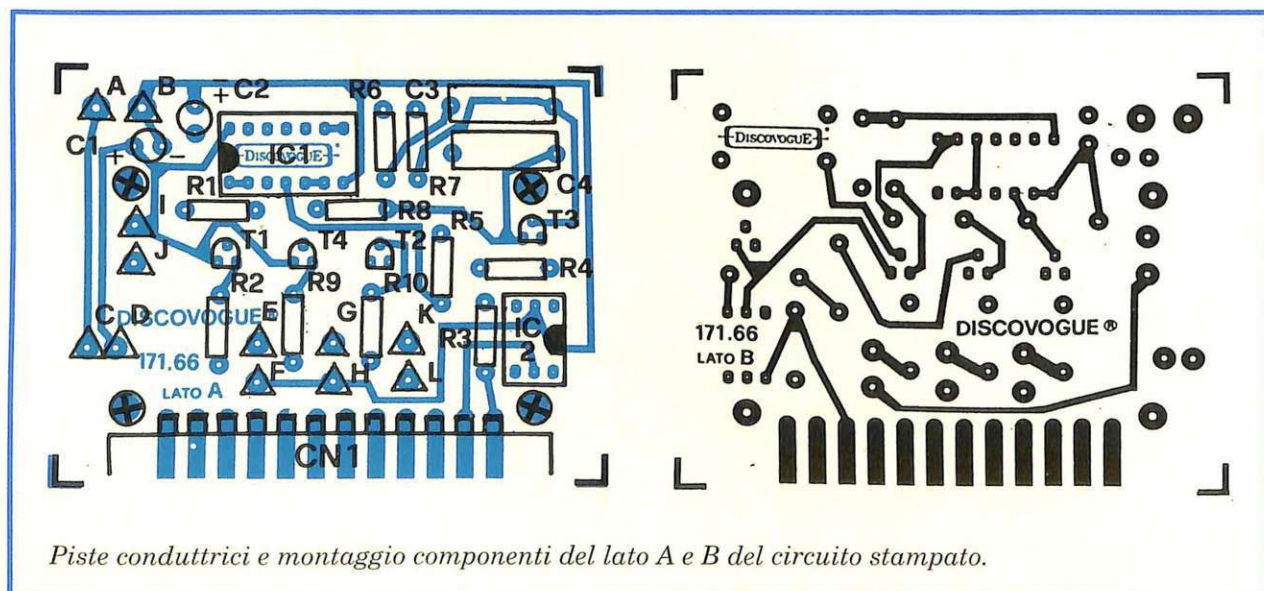
Oltre allo stagno e a un saldatore a stilo di medio-bassa potenza, serve solo un cacciavite per effettuare il fissaggio meccanico del circuito stampato al fondo del contenitore tramite le quattro apposite viti. Un buon lavoro potrà essere portato felicemente a termine osservando le classiche regole operative per i montaggi elettronici: trattare i componenti con cura (alcuni, come il fotoaccoppiatore IC2 e il connettore CN1, sono piuttosto delicati), effettuare saldature veloci con dosi di stagno adeguate e non eccessive, fare attenzione alle polarizzazioni dei condensatori elettrolitici (terminale positivo e negativo) e dei circuiti integrati (la tacca di riferimento indica sempre il primo pin, il numero 1 tra quelli disposti dual-in-line).

Si inizia montando il connettore CN1, saldando la prima fila dei 12 terminali al lato A e la fila degli altri 12 al lato B: l'operazione va eseguita lasciando il corpo del connettore il più possibile sporgente, in modo che a montaggio ultimato fuoriesca dal contenitore per essere agevolmente applicato alla porta utente del computer. Si continua montando sul lato A (e saldando sul lato B) prima i 12 chiodini capicorda ai punti contrassegnati dalle sigle A, B, C, D, E, F, G, H, I, J, K ed L, seguendo poi con i componenti più piccoli (resistenze, transistor), e terminando con i rimanenti più dimensionati, come i condensatori e i due circuiti integrati.

Ultimata la fase preliminare del montaggio, il circuito può essere inserito nell'apposito contenitore e fissato al fondo tramite le quattro piccole viti da inserire nei relativi pilastrini di sostegno: il connettore CN1 sporgerà dall'apertura anteriore.

Prima di chiudere il contenitore occorre ovviamente effettuare, tramite i cinque tranci di piattina bipolare a disposizione, tutti i collegamenti esterni. Innanzitutto si connette l'interruttore SW1 ai punti C e D; poi si procede con i due led L1 ed L2 (rispettivamente ai punti E-F e G-H).

Il potenziometro P1 ha tre terminali: quello di sinistra non va collegato, il cursore centrale va collegato al punto L, mentre il terminale destro va al punto K. La presa jack PS1 ha anch'essa tre terminali: quello centrale è il positivo da connettere al punto I, quello immediatamente vicino di destra è il segnale da collegare al punto J, mentre l'altro terminale non serve. Infine il cavetto di alimentazione con attacco per pila PS2 va allacciato ai punti A (filo rosso, positivo) e B (filo nero, massa).

Piste conduttrici e montaggio componenti del lato A e B del circuito stampato.

Elenco componenti

L'hardware Relaxer è composto in prevalenza da circuiti integrati, quindi componenti elettronici come resistenze e condensatori sono presenti in quantità relativamente contenute. Questo a garanzia di prestazioni di prim'ordine e affidabilità operativa. I limiti massimi di tolleranza si intendono 5 per cento per le resistenze e 10 per cento per i condensatori.

Semiconduttori (8)
(1) IC1: 4093 quad 2 input NAND Schmitt trigger
(1) IC2: 4N25 fotoaccoppiatore
(4) T1, T2, T3 e T4: BC547B
(1) L1: led cilindrico 5 mm. colore verde
(1) L2: led cilindrico 5 mm. colore rosso

Resistori (11)
(1) R1: 560 ohm 1/2 W
(1) R2: 120 Kohm 1/4 W
(5) R3, R4, R5, R6 ed R8: 1 Kohm 1/4 W
(2) R7 ed R10: 10 Kohm 1/4 W
(1) R9: 10 ohm 1/2 W
(1) P1: 1 Mohm potenziometro lineare

Condensatori (4)
(1) C1: 10 microF 16 VL elettrol. vert.
(1) C2: 4,7 microF 63 VL elettrol. vert.
(1) C3 e C4: 470 nanoF 100 VL poliest.

Vari (30)
(1) SW1: interruttore unipolare miniatura
(1) PS1: presa jack stereo 3,5 mm. da pannello
(1) S1: sensore di umidità superficiale cutanea a due anelli metallici, completo di cavetto bipolare di collegamento con spinotto jack stereo 3,5 mm
(1) PS2: cavetto di alimentazione 9 VL con attacco per pila 9 VL tipo transistor
(1) B1: pila 9 VL tipo transistor
(1) CN1: connettore 12+12 poli passo 3,96 mm
(1) circuito stampato cod. 171.66
(4) viti di fissaggio per c.s.
(12) chiodini terminali capicorda per c.s.
(5) tranci piattina bipolare lung. cm 15
(1) manopola con indice per potenziometro (particolari in colore rosso)
(1) contenitore plastico cod. 171.21

Terminate le operazioni di collegamento è consigliabile eseguire, tramite un tester, un semplice collaudo, verificando la presenza della 9 Volt c.c. sui punti A-B del circuito: ovviamente ciò deve essere fatto con lo strumento predisposto sulla corrente continua (10 volt f.s. c.c.) e collegando all'attacco PS2 la pila B1 a piena carica, accendendo l'interruttore SW1.

Occorre poi controllare il corretto funzionamento dell'oscillatore IC1a rilevando sul pin 3 un segnale di circa 3 Hz., valore determinato da R1 e C2. Il led verde L1 deve lampeggiare segnalando il regolare funzionamento dell'apparecchio. Se le misure danno esito positivo si può passare alla verifica finale pratica, collegando Relaxer alla porta utente del computer tramite il connettore CN1 che in parte fuoriesce dalla finestra anteriore del contenitore. Quindi si inserisce lo spinotto del sensore S1 nella presa PS1. Le suddette operazioni vanno ovviamente eseguite con gli apparecchi spenti, da accendersi solo successivamente.

Caricando il software nel computer e facendolo eseguire, si deve preventivamente effettuare la corretta taratura del sistema di rilevazione: inizialmente il potenziometro P1 Set va lasciato a metà corsa, verificando che il led rosso Contact si accenda e lampeggi velocemente ogni volta che c'è contatto tra anelli metallici del sensore e due dita di una mano (per esempio indice e medio). Poi si procede a una regolazione più accurata fintanto che, con soggetti in condizioni di buon relax, si raggiunge un lampeggio del led ad altissima frequenza ma che non sembri ancora acceso con continuità.

Il collaudo può richiedere tempo per la perfetta taratura di P1: questa particolare fase della procedura può tuttavia essere posticipata all'avvenuta installazione

del sistema. Se la verifica finale fornisce esito positivo, il circuito, già inserito nel fondo del contenitore, può essere definitivamente chiuso con il coperchio, in modo che alla fine i due led e il potenziometro sporgano correttamente dai corrispondenti fori presenti sul pannello frontale superiore, e che l'interruttore SW1 e la presa jack PS1 fuoriescano dai fori laterali di sinistra (PS1 dal penultimo in fondo, SW1 dal secondo anteriore). Il cavetto di alimentazione PS2 con la pila B1 deve invece uscire dal foro presente sul retro del coperchio.

Inserimento e fissaggio del circuito nel contenitore plastico. cod. 171.21

Il potenziometro, la presa jack e l'interruttore di accensione sono dotati di sporgenze filettate a cui applicare i relativi anelli metallici di fissaggio. Si completa l'installazione fissando la manopola all'alberino del potenziometro in modo che, ruotandolo tutto in senso antiorario, l'indice di posizione si orienti a sinistra in basso, cioè sul minimo della frequenza-base di impulso.

Una volta pronto, Relaxer va usato con la massima cautela, in ambienti tranquilli, solo su soggetti non impressionabili e senza problemi di salute o di malattie nervose, escludendo sempre e comunque bambini e anziani.

Sono possibili sia il rilassante training autogeno che il più divertente lavoro in coppia (soggetto analizzato e utente che lo mette alla prova con quesiti anche imbarazzanti). Ovviamente l'apparecchio fornisce i risultati migliori se usato solo per raggiungere condizioni di relax in modo graduale e senza pretese di leggere subito su video responsi sensazionali. Soggetti particolarmente tesi o emozionati possono impiegare anche ore per calmarsi: l'uso continuato e metodico di Relaxer aiuta ad abbreviare questi tempi.

Software di funzionamento

Questo programma permette un immediato ed efficace uso del dispositivo elettronico Relaxer fornendo un completo responso grafico-quantitativo sullo stato di rilassamento psico-fisico di un qualsiasi soggetto sottoposto ad analisi mediante un apposito bisensore a clip applicato a due dita della mano.

Prima di (e durante) ogni procedura di misurazione è possibile memorizzare un valore standard di riferimento delle condizioni della persona esaminata: questo consente un monitoraggio veritiero, continuo e aggiornato in tempo reale. La visualizzazione del responso avviene su un range grafico a ben 27 livelli, e può variare da un limite estremo (Relax Totale) a quello opposto (Massima Tensione), passando attraverso una soglia centrale d'indicazione (Set).

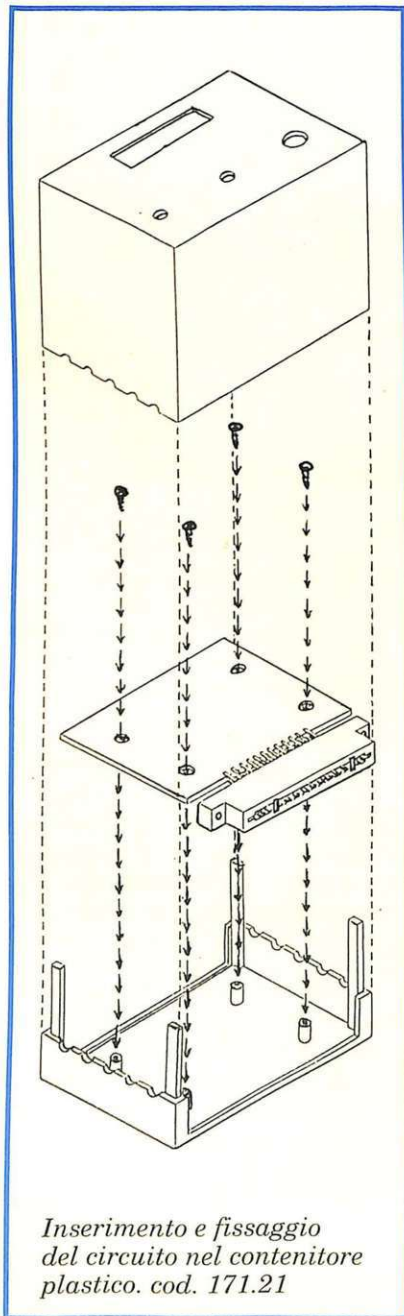
In pratica il software è in grado di elaborare il segnale trasmesso, attraverso l'input PB0 della porta utente del computer, dal circuito di rilevazione parametrica e di generazione a frequenza variabile degli impulsi elettronici.

Terminata l'operazione di caricamento, non appena il programma parte, compare la videata-monitor a fondo e bordo di colore nero.

Nella parte superiore dello schermo sono presenti il nome Relaxer 2 (in vari colori) e il data-set di copyright (in colore grigio). L'area-video rimanente viene suddivisa in due distinti settori: quello inferiore contiene un prezioso memorandum, stampato in colore verde, che invita l'utente a tarare in via preliminare il potenziometro Set dell'apparecchio Relaxer prima di procedere a ogni elaborazione.

In pratica occorre regolare la manopola in modo che vengano prodotti impulsi elettronici a frequenza abbastanza elevata da permettere un'analisi affidabile: a tal scopo il led rosso Contact dell'apparecchio Relaxer deve lampeggiare molto velocemente (ma senza sembrare acceso con continuità) al contatto tra sensori e dita della mano del soggetto analizzato.

Il settore superiore contiene due riquadri: quello di destra, in giallo, è il display di visualizzazione grafica a 27 livelli del responso fornito in tempo reale.

Un piccolo segnalino di colore bianco, inizialmente posto sulla soglia centrale di set, reagisce subito all'arrivo dei primi impulsi trasmessi dal Relaxer, schizzando verso destra (limite di Massima Tensione) indipendentemente

Come si acquista il materiale già pronto

È disponibile la versione hardware, ovvero l'apparecchio già montato, collaudato e funzionante, completo di istruzioni di installazione e uso. Codice 171.00, lire 87.000. Chi ha un minimo di esperienza con l'elettronica e col saldatore può acquistare la versione Hardware Kit, una scatola di montaggio completa comprendente, oltre a tutto il materiale indicato nell'elenco componenti, anche le istruzioni di assemblaggio, collaudo, installazione e uso. Codice 171.10, lire 74.500. È inoltre possibile richiedere il Personal Set, una confezione comprendente solo il circuito stampato, il connettore, il contenitore, i relativi accessori di fissaggio e le istruzioni di assemblaggio, collaudo, installazione e uso, per costruire l'apparecchio Relaxer avendo già a disposizione il rimanente materiale necessario. Codice 171.20, lire 47.500.

Per quanto riguarda il software esistono diverse versioni di programmi per Commodore 64 (naturalmente C 128 compatibili), con caratteristiche variabili in base alle prestazioni richieste e al tipo di supporto di memorizzazione utilizzabile (audio-cassetta o dischetto):

- ***Software dimostrativo su cassetta:** semplice, ma versatile e potente, permette di analizzare i dati relativi alle condizioni psico-fisiche di una qualsiasi persona sottoposta a rilevazione, fornendo un appropriato messaggio-responso (scelto tra i cinque previsti) di chiara e immediata interpretazione. Possibilità di memorizzazione istantanea e continua di valori di riferimento. Codice 171.61, lire 11.000.*
- ***Software di funzionamento su cassetta:** stesse caratteristiche del programma cod. 171.61, ma con videata grafica multicolor di supporto e routine supplementare per la proiezione grafica su un display a 27 livelli del responso fornito. Codice 171.62, lire 27.000.*
- ***Software di funzionamento su dischetto:** stesse caratteristiche del programma cod. 171.62, ma con possibilità di visualizzare e memorizzare una sequenza statistica di ben 50 rilevazioni. Codice 171.63, lire 37.000.*

Tutti gli ordini d'acquisto possono essere effettuati tramite lettera, indirizzando in busta chiusa a: Discovogue, P.O. Box 495, 41100 Modena.

I prezzi si intendono Iva compresa, con pagamento contrassegno e spese di spedizione a carico del destinatario. Gli invii si effettuano ovunque, entro 24 ore dall'arrivo dell'ordine, tramite pacco postale che, a richiesta, può essere anche urgente (con maggiorazione delle spese aggiuntive). Ogni ordine dà diritto a ricevere in omaggio, oltre a una gradita sorpresa, anche la Mailing Card personalizzata e codificata che consente di ottenere sconti e agevolazioni in eventuali ordini successivi.

dalle reali condizioni psico-fisiche della persona al momento a contatto col bi-sensore. Occorre attendere una decina di secondi e quindi, come avverte il riqudro settoriale di sinistra (in rosso chiaro) procedere alla memorizzazione del valore di riferimento: l'operazione è istantanea, basta premere il tasto-funzione F1 e il segnalino si porta al centro del display (Set).

A questo punto, a ogni crescere emozionale del soggetto corrisponderà uno spostamento verso destra del segnalino, mentre a ogni rilassamento psico-fisico farà seguito un rapido ritorno alla soglia, o addirittura un movimento verso sinistra (fino all'eventuale estremo Relax Totale): i cicli di elaborazione si verificano ogni 15 decimi di secondo, e dunque con tale frequente periodicità si procede agli aggiornamenti.

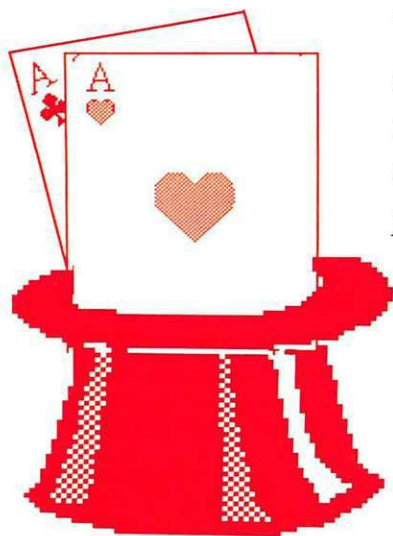
La memorizzazione di un valore può avvenire in qualsiasi momento, anche ripetutamente: l'importante è ricordare che il responso viene dato in base a quanto il computer legge sull'input al momento di questa importantissima operazione, da farsi dunque inizialmente solo quando la persona analizzata si è stabilizzata interiormente, e da ripetersi ogniqualvolta il cursore del display si sposta troppo su e giù per i 27 livelli senza stabilizzarsi.

È possibile il training autogeno, per tentare di portarsi in condizioni di assoluto relax da soli, pensando al nulla, o al colore verde, o a un paesaggio di tranquilla campagna, oppure per operazioni di self-emotion pensando a scene, d'azione o di movimento.

I risultati migliori si ottengono però quando si usa Relaxer in due: uno è il soggetto analizzato che subisce, e l'altro è colui che lo importuna con domande imbarazzanti, o stimola i suoi sensi con rumori (udito), abbagli di luce (vista), solletico (tatto), verificando poi via software l'alterazione psico-fisica più o meno evidente.

In ogni caso è importante operare in un ambiente tranquillo, il più possibile esente da fattori esterni che possano alterare indirettamente le condizioni di chi viene sottoposto a sondaggio: basta a volte un rumore come quello di una porta che sbatte per provocare un'impennata emozionale (e una conseguente scomparsa di relax faticosamente raggiunto).

Durante il funzionamento del programma anche il dispositivo Relaxer svolge un'appropriata azione di monitoraggio assai utile all'utente, tramite le segnalazioni dei led di colore verde (Power) e rosso (Contact) che segnalano, rispettivamente, il corretto funzionamento dell'apparecchio e un avvenuto contatto tra bi-sensore di rilevazione e dita della mano della persona sottoposta ad analisi.

Daniele Malavasi

Vi piacerebbe volare su un armatissimo aereo da caccia? Ora finalmente crederete di farlo davvero davanti al vostro monitor con questo magnifico programma di simulazione di guerra!

Coppia d'assi

Malgrado l'impegno verso un totale disarmo nucleare, gli armamenti bellici delle due super potenze nell'anno 1995 erano ancora molto ingenti. Anzi, si stava per perfezionare da ambo le parti una bomba dalle particolari caratteristiche; chi l'avesse ultimata per primo, con poca fatica si sarebbe impossessato dell'intero globo terrestre.

Ognuna delle due superpotenze, tramite lo spionaggio militare, conosceva il livello raggiunto dall'altra, che pressapoco equivaleva al proprio. Da entrambe le parti si pensava alla eventuale soluzione da approntare nel caso che la potenza avversaria ultimasse per prima il progetto.

Entrambe le superpotenze decisero allora di mandare un caccia dell'ultima generazione a distruggere i laboratori avversari. Uno dei due aerei fu dotato di un comando computerizzato e di un validissimo pilota, il tenente Mark Ace.

In tutta segretezza, i velivoli furono preparati, armati di una sufficiente quantità di missili e fatti partire.

Dopo varie ore di volo a bassa quota, il tenente diede un'occhiata al radar e scorse uno strano puntino luminoso in avvicinamento. Dotato di nervi saldissimi, capì subito di che cosa si trattava: armò la mitragliera e i missili a ricerca di calore e si mise in assetto di combattimento. Sapeva che poco dopo sarebbe scoppiato il finimondo. Confidava nella sua abilità e... in un pizzico di aiuto da parte vostra!

Il simulatore prevede anche due giocatori ognuno dei quali controlla uno dei due velivoli cercando di intercettare e abbattere quello dell'avversario.

Come si gioca

A comandare l'aereoplano siete proprio voi, e non pensiate sia uno scherzo! Il gioco comincia a terra, dove si vede inquadrato il vostro aereo. Premendo il bottone del joystick, l'aereo decolla. Muovendo il joystick nelle varie direzioni, potrete compiere qualsiasi tipo di manovra: cabrata, volo rovesciato, picchiate, eccetera.

La prima difficoltà che incontrerete è senz'altro l'uso simultaneo di joystick e tastiera. Quest'ultima vi permetterà di controllare la potenza del motore e quindi la velocità dell'aereo, la mitragliera o i missili teleguidati e la mappa con la posizione dei due aerei.

I comandi, che all'inizio sembrano molto complicati, a chi ha

un po' di pazienza per imparare a usarli si rivelano poi abbastanza facili.

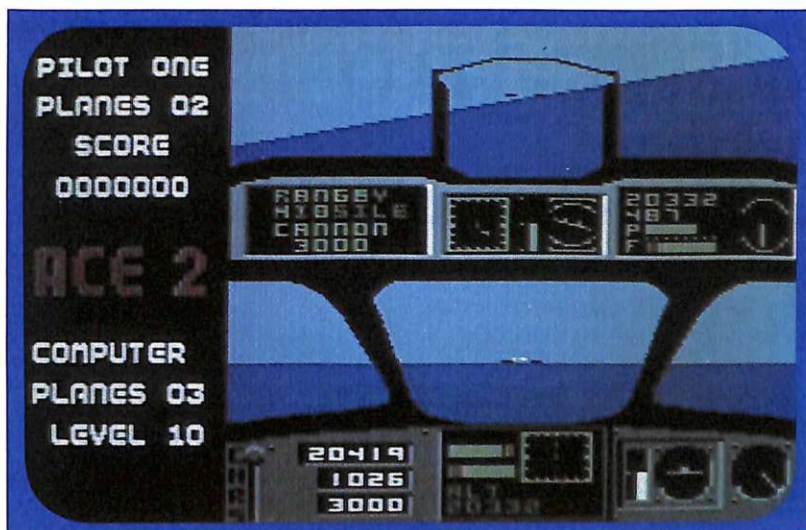
La presentazione del gioco è molto curata, sia per la grafica sia per il sonoro. La musica di sottofondo introduce e ambienta bene il gioco.

Subito dopo la presentazione appaiono sullo schermo tutte le opzioni disponibili: il numero di aerei di scorta, i colpi a disposizione di ognuno di essi, il detector anti-distruzione e la possibilità di scegliere uno o due giocatori. Nel caso scegliate un solo giocatore, anche il livello di difficoltà sarà da 1 a 20. Ognuno di essi si seleziona con il joystick.

Prima di far iniziare la partita, dovrete selezionare il tipo di missione: ne esistono due, la prima consiste nel distruggere l'aereo nemico, mentre la seconda prevede la distruzione della base di ricerche avversaria. Facendo partire la missione, lo schermo si divide orizzontalmente in due parti e ognuna di queste rappresenta la cabina di pilotaggio e tutti gli strumenti di volo.

Dopo un brevissimo decollo, dovrete osservare una serie di dettagli. Per ogni aereo, vi è un radar e uno strumento che indica, istante per istante, l'assetto del velivolo. Due barrette colorate segnalano la potenza erogata dal motore e il carburante ancora a disposizione. La quota, il punteggio e gli aerei rimasti sono segnalati con strumenti separati, mentre il computer di bordo segnala a intermittenza sul pannello i colpi a disposizione per la mitragliera, il pericolo imminente, la quota troppo bassa, tipi di missili a disposizione eccetera.

Ma vediamo più da vicino i comandi: per il pilota uno, la potenza del motore si seleziona con i tasti S e F, mentre per il secondo si seleziona con H e K. La X e la V servono rispettivamente ai due giocatori per far uscire la mappa della zona.

I tasti più importanti sono invece E e U, che servono a selezionare il tipo di fuoco. Se sullo schermo appaiono due trattini significa che avrete armato la mitragliera e il computer vi segnalerà i colpi ancora a disposizione; se non appare niente oppure un riquadro tratteggiato significa che avete armato i missili aria-aria o aria-terra. Prima del decollo dovrete infatti sceglierli in funzione della missione e installarli sotto le ali.

Durante il combattimento è possibile vedere la mappa della zona semplicemente premendo il tasto X per il primo giocatore e il tasto V per il secondo.

La grafica è ben congegnata e dà davvero l'impressione di essere dentro un simulatore; gli effetti sonori, senza essere strabilianti, servono bene allo scopo. Se un aereo viene colpito e distrutto, il successivo ripartirà dal punto iniziale con lo stesso armamento del precendente (modificabile a vostra scelta).

Se avrete totalizzato un punteggio dignitoso, entrerete in una mini classifica formata da cinque record.

Consigli e conclusioni

Per entrare in confidenza con questo gioco occorre senz'altro un po' di tempo, ma esso non tarderà a darvi parecchie soddisfazioni, soprattutto quando giocherete in due.

Vi consigliamo di selezionare il modo 2 Player all'inizio, e di provare a eseguire manovre senza cominciare subito il combattimento.

Se dopo un po' di pratica pensate di essere diventati bravi, lanciate pure la vostra sfida al computer: constaterete però che è facile essere abbattuti!

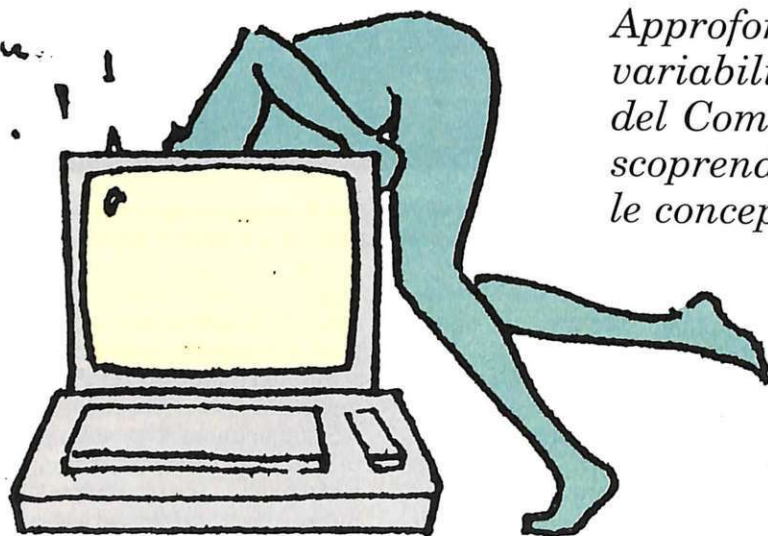
Una delle più grandi difficoltà del combattimento è quella di scappare dall'inseguimento di un missile dopo essere stati puntati.

I modi validi per scamparla sono due: dare tutta potenza ai motori e salire in verticale cercando di descrivere un cerchio, mantenendosi così il più possibile lontano dai missili (lo si può vedere nel radar); oppure fare l'operazione contraria, e cioè spegnere i motori e scendere in picchiata compiendo le evoluzioni più strane (questo secondo metodo è valido naturalmente solo se siete ad alte quote).

Un altro consiglio: tenete sotto controllo il computer di bordo in ogni momento, eviterete così di schiantarvi per una banale distrazione. Un segnale sonoro, comunque, vi avvertirà del pericolo.

Non ci resta che augurarvi buon divertimento e buona caccia!

Oscar Maeran

L'arcano delle variabili

Approfondiamo l'analisi delle variabili numeriche del Basic del Commodore 64 scoprendo come il computer le concepisce e le rappresenta.

Nel manuale d'uso del C64 vi sono solo pochi accenni alle variabili numeriche, cioè a quei simboli che rappresentano le locazioni di memoria nelle quali viene immesso il valore assegnato alle variabili stesse. Il manuale specifica che esse si distinguono in variabili intere ed in variabili in virgola mobile e che le prime sono contrassegnate dal carattere %. Al riguardo non dice molto di più.

Chi ormai da tempo ha familiarizzato con il C64 o ha avuto occasione di leggere testi che lo riguardano, sa come il calcolatore tratti i numeri attribuiti alle variabili, ma chi è alle prime armi può incontrare qualche difficoltà a comprendere la loro gestione.

Infatti, l'aggettivo intere potrebbe far pensare che tali variabili recepiscano qualsiasi numero, purché intero, mentre quelle in virgola mobile possano ricevere solo numeri decimali.

Invece non è così: il sistema del C64 consente di assegnare alle variabili intere solo numeri interi compresi fra -32768 e +32767 e a quelle in virgola mobile numeri interi o decimali compresi fra più o meno 2.93873588E-39 e più o meno 1.70141183E+38.

Per esempio, se proviamo a digitare in modo diretto:

```
A%=40000:PRINTA%
```

uscirà ?Illegal Quantity Error perché 40000 è fuori dei limiti consentiti per le variabili intere; se invece scriviamo:

```
A=40000:PRINT A
```

il computer giustamente visualizzerà 40000. Qualora poi provassimo ad assegnare a una variabile intera un numero decimale, essa recepirebbe solo la parte intera, ed è per questo che in taluni casi viene usata in sostituzione della funzione INT. Per esempio:

```
B=125.3:A%=B:PRINTA%
```

ha gli stessi effetti di:

```
B=125.3:A=INT(B):PRINT A
```

poiché in entrambi i casi A% e A assumono come valore 123.

Vediamo ora di esaminare come il C64 gestisce i valori attribuiti alle variabili numeriche, sia intere sia in virgola mobile, facendo astrazione da ciò che avviene nel sistema operativo e limitandoci a guardare nelle locazioni della memoria RAM riservata ai programmi Basic, ossia all'utente, ove vengono inseriti i dati delle variabili.

Le variabili numeriche occupano sempre sette byte, e ciò vale tanto per quelle intere quanto per quelle in virgola mobile. Per guardare ciò che avviene in questi byte useremo altri due byte che fanno parte dei cosiddetti puntatori, cioè di quei byte che lavorano in coppia e che servono a raggiungere determinate locazioni.

I byte che fanno al caso nostro sono il 45 e il 46. I valori che essi contengono permettono di arrivare alle locazioni delle variabili e di rilevarne il contenuto tramite la funzione PEEK.

Variabili intere

Quando la variabile è intera, i sette byte hanno le seguenti funzioni:

- byte 1: contiene il codice ASCII del primo carattere della variabile più 128;
- byte 2: contiene il codice ASCII dell'eventuale secondo carattere della variabile più 128; se la variabile ha un carattere solo, il valore del byte è 128;
- byte 3 e 4: contengono i valori relativi al numero introdotto, che si ottiene moltiplicando per 256 il valore del byte 3 e aggiungendo quello del byte 4;
- byte 5, 6, 7: non vengono utilizzati dalle variabili intere, perciò contengono sempre zero.

Controlliamo se è vero e per farlo digitiamo il **listato 1**, seguendo la numerazione corrispondente al listato dell'analogo programma presente nella cassetta acclusa alla rivista, integrato da numerose linee di commento.

Diamo il RUN e inseriamo 40000. Esce, a conferma di quanto già detto, ?Illegal Quantity Error in 120, poiché abbiamo immesso un numero non compreso fra -32768 e +32767. Proviamo con 15821; usciranno i seguenti numeri rappresentanti rispettivamente il contenuto dei byte da 1 a 7: 193 - 128 - 61 - 205 - 0 - 0 - 0, che ci consentono di rilevare:

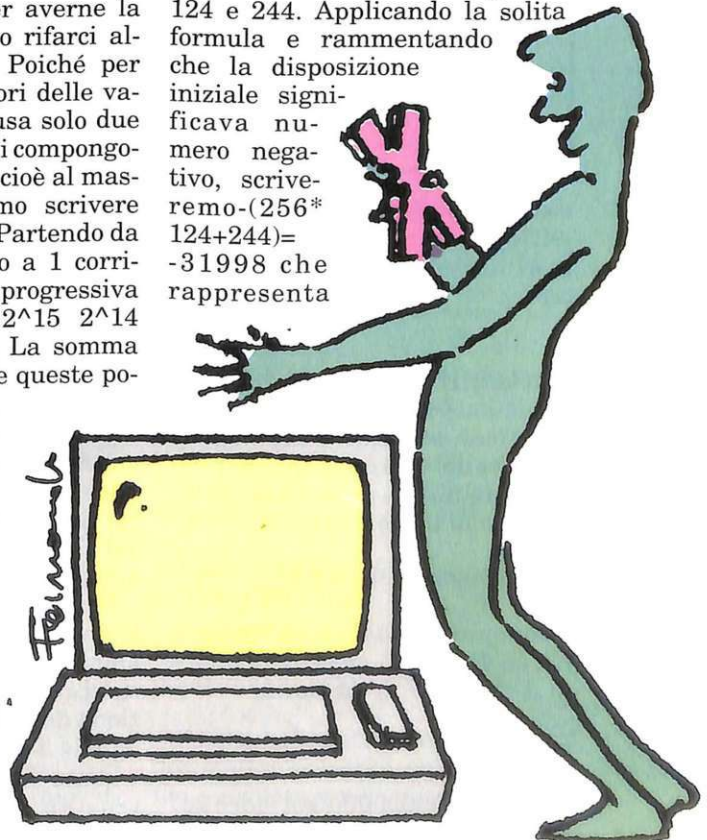
- byte 1 : 193-128=65, codice ASCII di A, lettera che abbiamo usato per indicare la nostra variabile;
- byte 2 : 128-128=0, in quanto la variabile è individuata da un carattere solo;
- byte 3 : 61, valore da moltiplicare per 256;
- byte 4 : 205, valore da aggiungere al risultato dell'operazione di cui sopra per risalire al numero introdotto;
- byte 5, 6, 7 : hanno valore nullo perché non vengono utilizzati.

Eseguiamo 256*61+205 ed avremo 15821. Abbiamo dunque visto come il C64 memorizza nella RAM le variabili intere ed i valori ad esse assegnati.

Facciamo la prova con un numero negativo, per esempio -15. Avremo, a fronte dei rispettivi byte, i seguenti numeri: 193 - 128 - 255 - 241 - 0 - 0. I primi due, rispetto a prima, sono invariati perché invariata è l'identificazione della variabile. Usiamo il terzo e il quarto per calcolare256*255+ 241, il cui risultato è 65521, che non ha apparentemente a che fare con il numero introdotto. Ma quando il numero introdotto è negativo, occorre togliere 65536. Eseguendo 65521-65536 otteniamo -15, che è ciò che cercavamo.

Cerchiamo adesso di chiarire perché i limiti entro cui possono operare le variabili intere siano -32768 e +32767. Per averne la spiegazione dobbiamo rifarci all'aritmetica binaria. Poiché per immagazzinare i valori delle variabili intere il C64 usa solo due byte, se tutti i bit che li compongono vengono posti a 1, cioè al massimo valore, possiamo scrivere 11111111 11111111. Partendo da destra, ogni bit posto a 1 corrisponde a una potenza progressiva di 2, vale a dire: 2^15 2^14 2^13....2^2 2^1 2^0. La somma dello sviluppo di tutte queste potenze è pari a 65535. Ma l'ultimo bit a sinistra ha una funzione particolare poiché viene utilizzato per stabilire il segno del numero. Se tale bit è a zero, significa che il numero è positivo, se è a uno, significa che è negativo. Perciò quando il numero è positivo ai fini della determinazione del suo valore concorrono 15 bit e non 16. La somma del valore di 15 bit posti a 1 è uguale a 32767, che costituisce il massimo numero positivo rappresentabile da una variabile intera. Se tutti i bit sono a 1, compreso il bit più a sinistra, questo avvisa che il numero è negativo. In questo caso per risalire dal numero in notazione binaria a quello in notazione decimale occorre:

- rammentare che il risultato delle operazioni da eseguire dovrà essere considerato negativo;
- togliere 1 dal bit più a destra, ricorrendo al riporto se non v'è capienza;
- dopo di che porre a zero i bit a uno e a uno i bit a zero.

Poniamo di avere i due bit nella forma 10000011 00001100, rispettivamente pari a 131 e 12. Togliendo 1 dal bit più a destra e ricorrendo al riporto abbiamo 10000011 00001011. Invertendo, otteniamo 01111100 11110100. I valori decimali sono ora uguali a 124 e 244. Applicando la solita formula e rammentando che la disposizione iniziale significava numero negativo, scriveremo-(256* 124+244)= -31998 che rappresenta

il valore dei bit in origine.

Visto il metodo da usare con i numeri negativi, supponiamo di avere i due byte nei termini 10000000 00000000. Il bit più a sinistra indica che si tratta di un numero negativo. Perciò dobbiamo togliere 1 dal bit di destra. Eseguiamo ricorrendo al riporto sino a che troviamo capienza. Otterremo 01111111 11111111. Quindi invertiamo e scriviamo 10000000 00000000 (il che in questo caso particolare riconduce i byte a come erano in origine). Ora applichiamo la formula e rammentando che il risultato deve essere negativo, calcoliamo -(256*128+0)=-32768. Poiché non c'è altra forma iniziale dei due byte che possa andare oltre questo risultato, abbiamo conferma che -32768 è il limite inferiore dei numeri inseribili nelle variabili intere.

Listato 1.

```
110 PRINTCHR$(147)
120 PRINT"INSERISCI UN NUMERO INTERO:":INPUTA%:PRINT
130 PRINT"CONTENUTO DEI BYTE DELLA VARIABILE"
140 PRINT:"INTERA:":PRINT
150 P=256*PEEK(46)+PEEK(45)
160 FORK=1TO7
170 PRINT"BYTE";K;":";PEEK(P+K-1):NEXT:PRINT
500 REM SE BYTE 3 >= 128, TOGLIERE 65536
520 REM DALLA FORMULA 256*BYTE3+BYTE4
540 IFPEEK(P+2)>=128THEN580
550 PRINT"256 *";PEEK(P+2);"+";PEEK(P+3);"=";256*PEEK(P+2)+PEEK(P+3)
560 GOTO590
580 PRINT"256 *";PEEK(P+2);"+";PEEK(P+3);" - 65536 = ";
585 PRINT(256*PEEK(P+2)+PEEK(P+3))-65536
590 PRINT:PRINT"CHE È IL NUMERO IMMESSO NELLA VARIABILE"
```

Variabili in virgola mobile

Iniziamo ricordando l'uso della notazione scientifica dei numeri da parte del C64 quando superano una deteminata grandezza. Se digitiamo in modo diretto:

A=999999999:PRINT A

il computer visualizzerà 9999999 99; se digitiamo, mentre il valore di A permane in memoria,

PRINT A+1

anziché 1000000000 sul video apparirà 1E+09. Non potendo visualizzare più di nove cifre, il C64 utilizza la notazione standard da 0 a 999999999 (tanto per i numeri positivi che per i negativi), dopo di che passa alla notazione scientifica.

La notazione scientifica si avvale del simbolo E+ o -X, che vuol significare che il numero di sinistra E va moltiplicato (se c'è il segno di addizione), oppure va diviso (se c'è il segno di sottrazione) per 10 elevato a X.

All'inizio abbiamo detto che il massimo numero positivo introducibile in una variabile in virgola mobile è +1.70141183E+38. In notazione normale, moltiplicando per 10^38 si dovrebbe scrivere 1701411830000000 00 00 000 000 00 0000000000. È un numero assai grande, ma è sufficiente dover calcolare un fattoriale per far apparire il solito Overflow. Se poniamo A uguale a tale numero digitato per esteso e poi diamo il comando PRINT A, sul video apparirà trasformato in notazione scientifica.

Il numero prima della E e quello dopo sono rispettivamente denominati Mantissa (M) e Caratteristica (C), per cui nella numerazione decimale ogni numero sarà uguale a M * 10^C.

Tutto ciò premesso andiamo a vedere cosa succede nei sette byte che il C64 utilizza per immettervi i valori rappresentati dalle variabili numeriche, quando la variabile è in virgola mobile. Diamo qui un elenco delle funzioni di detti byte, che si differenziano alquanto rispetto al caso delle variabili intere:

- byte 1: codice ASCII del primo carattere della variabile (non più aumentato di 128 come per le variabili intere);
- byte 2: codice ASCII, come sopra, per il secondo carattere. Se la variabile ha un carattere solo, il contenuto è zero;
- byte 3: valore utile per individuare la Caratteristica, cioè l'esponente. Poiché gli esponenti possono essere positivi o negativi e i numeri in un byte possono variare solo da 0 a 255, il sistema del C64 prevede che l'esponente 0 sia uguale a 128, per cui in binario gli esponenti positivi vanno da 128 a 255 e quelli negativi da 0 a 127. Ne consegue che per ottenere l'esponente in base 2 occorre togliere 128 dal valore contenuto nel byte 3;
- byte 4, 5, 6 e 7: contengono i valori che costituiscono la Mantissa.

Con procedimento analogo sostanzialmente a quello usato per le variabili intere, al fine di esaminare il contenuto di detti byte, digitiamo il **listato 2** dopo aver dato il New e continuando il collegamento con i numeri di linea del listato in cassetta, quindi inseriamo nuovamente il numero 15821, come già abbiamo fatto con il programma per le variabili intere. Il numero introdotto è lo stesso, ma avendo utilizzato questa volta una variabile in virgola mobile, i

Listato 2.

```
690 PRINTCHR$(147)
700 PRINT"INSERISCI UN NUMERO:":INPUT A
710 PRINT"VALORI DECIMALI NEI SETTE BYTE:"
720 P=256*PEEK(46)+PEEK(45):PRINT
730 FORK=0TO6
740 PRINT PEEK(P+K);" ";:NEXT:PRINT
```

valori che compaiono nei byte sono diversi, e precisamente: 65 - 0 - 142 - 119 - 52 - 0 - 0. Analizziamoli:

Listato 3.

```
830 FORK=1TO5:M(K)=PEEK(P+K+1):NEXT:REM IMMAGAZZINA IN M(K) GLI ULTIMI 5 BYTE
840 PRINT:PRINT"QUATTRO BYTE DELLA MANTISSA IN BINARIO:"
860 FORK=2TO5
870 BY$(K)="":FORJ=7TO0STEP-1
880 IFINT(M(K)/2^J)=0THEN900
890 BY$(K)=BY$(K)+"1":M(K)=M(K)-2^J:GOTO910
900 BY$(K)=BY$(K)+"0"
910 NEXT:NEXT
920 PRINTBY$(2),BY$(3),BY$(4),BY$(5)
990 PRINT:PRINT"GLI STESSI RRIUNITI IN UNICA STRINGA:"
1010 S$=BY$(2)+BY$(3)+BY$(4)+BY$(5):PRINTS$
```

• 65.= codice ASCII della lettera A della variabile;

• 0 = zero, perché la variabile è contrassegnata da una lettera sola;

• 142 = valore che individua la Caratteristica. Per operare in binario e ottenere l'esponente in base 2 dobbiamo togliere 128 e in questo caso avremo: esp. = 142 - 128 = 14;

• 119 - 52 - 0 - 0 = valori che determinano la Mantissa.

Conosciuti i valori contenuti nei byte, vediamo di risalire da essi al numero introdotto, cioè a 15821. Trasformiamo dapprima i valori dei byte che concernono la Mantissa in una sequenza binaria. Daremo per comodità di programmazione a tale sequenza la forma di stringa. Perciò, a quanto già abbiamo digitato, aggiungiamo le linee del **listato 3**.

Anche per le variabili in virgola mobile occorre distinguere se il numero introdotto è positivo o negativo. La struttura del C64 prevede che il primo bit a sinistra del primo byte sia inizialmente sempre posto a 1.

Se il numero introdotto è negativo, lo lascia com'è. Se è positivo lo fa diventare 0. Reintroduciamo 15821, e poiché è positivo, certamente saràposto a 0 il primo bit a sinistra. Infatti la stringa che riunisce tutti i bit si presenta così: 01110111001101000000000000000000. Quando c'è un mutamento, dobbiamo riportare a 1 il primo bit per avere l'effettivo controvalore in binario. Prima, però, prendiamo nota in una variabile della positività o negatività del numero. Provvediamo a questo con:

```
1040 R=1+2*(LEFT$(S$,1)="1")
```

per cui sappiamo che, se il primo carattere della stringa è uguale a "1", allora R è uguale a -1, se il primo carattere della stringa è diverso da "1", allora R è uguale a +1. Il valore di R ci servirà in seguito. Possiamo ora modificare la stringa con:

```
1050 S$="1"+RIGHT$(S$,31)
```

mettiamo a 1 il primo bit. Nel caso in cui il numero introdotto fosse stato negativo, il computer l'avrebbe lasciato a 1 e anche la linea 1050 lo avrebbe lasciato immutato.

Continuando nel proposito di risalire al nostro numero 15821, digitiamo le linee del **listato 4**. L'esponente E fornisce due indicazioni. Se E è positivo, ci dice quanti bit dobbiamo prelevare a sinistra della stringa per avere la parte intera, sia che il numero sia intero o che sia decimale. Se E è negativo, ci dice quanti zeri dobbiamo aggiungere a sinistra della stessa.

Per questa seconda evenienza prepariamo una stringa con 128 zeri, dato che E negativo può giungere sino a 128:

```
1310 Q$="0":FORK=1TO127:Q$=Q$+"0":NEXT
```

Poi teniamo presente che:

• se E=0, allora la parte intera è zero e quella decimale è data da tutta la stringa;

• se E<0, allora la parte intera è zero e quella decimale è data dalla stringa più E zeri;

• se E > del numero dei bit della stringa, allora alla stessa vanno aggiunti tanti zeri (prelevati da Q$) quanti ne occorrono per avere una nuova stringa di E bit; quindi come visibile nel **listato 5**.

Fuori dai suddetti casi particolari, la parte intera è determinata dagli E bit a sinistra della stringa, ossia:

```
1350 IN$=LEFT$(S$,E)
```

Se E è maggiore o uguale al numero dei bit presenti nella stringa, è ovvio che non esiste parte decimale, perciò:

```
1360 IF E >=LEN(S$) THEN DC$="0":GOTO1380
```

Altrimenti la parte decimale è data dai bit rimanenti a destra della stringa, cioè:

```
1370 DC$=RIGHT$(S$,(LEN(S$)-E))
```

ed infine possiamo scrivere:

```
1380  PRINT:PRINT"BIT  DELL'INTE-
RO:":PRINT IN$
1400 PRINT"BIT DEI DECIMALI:":PRINT DC$
```

Giunti a questo punto, reintroduciamo ancora una volta 15821. Avremo i seguenti ulteriori risultati:

Esponente = 14

Bit dell'intero:
11110111001101

Bit dei decimali:
000000000000000000

Poiché nella fattispecie i bit dei decimali sono tutti a zero, dobbiamo considerare solo quelli dell'in-

Listato 4.

```
150 PRINT:PRINT"LA STESSA STRINGA DOPO LA
    VERIFICA DEL"
1160 PRINT"SEGNO EFFETTUATA SUL PRIMO BIT"
1180 PRINTS$
1290 E=M(1)-128:PRINT:PRINT"ESPONENTE= ";E
```

Listato 5.

```
1320  IF E=0 THEN IN$="0":DC$=S$:GOTO1380
1330  IF E<0 THEN IN$="0":DC$=LEFT$(Q$,-
      E)+S$:GOTO1380
1340  IF E > LEN(S$) THEN S$=LEFT$(S$+Q$,E)
```

tero. Ma i bit dell'intero, così come si presentano, non ci sono ancora d'aiuto. Occorre immaginarli come una serie di byte costruita a partire da destra a cui eventualmente manchino a sinistra gli zeri necessari affinché il numero complessivo dei bit sia un multiplo di 8.

Prima di incaricare il computer di proseguire nelle operazioni, proviamo ad eseguire personalmente i conteggi. Perciò ricaveremo dai bit dell'intero indicati sul video i seguenti byte 00111101 11001101, ottenuti aggiungendo a sinistra gli zeri occorrenti alla loro completezza.

I loro rispettivi valori in decimale sono 61 e 205. Eseguendo 256*61+205 otteniamo 15821 che è il numero a suo tempo introdotto.

Facciamo ancora una prova immettendo in input lo stesso numero, ma negativo, vale a dire -15821. Noteremo sul video che i sette byte contengono 65 - 0 - 142 - 247 - 52 - 0 - 0. È cambiato il valore nel quarto byte perché il suo primo bit, posto inizialmente sempre a 1 dal C64, è rimasto tale in quanto il numero introdotto è negativo. Pertanto anche l'operazione della linea 1050 non ha avuto effetto e ne abbiamo conferma confrontando sullo schermo le due stringhe che riuniscono i quattro byte. Gli altri byte risultano invariati rispetto alla prova precedente, ed immutati sono anche i bit dell'intero e quelli dei decimali. Per il fatto che il primo bit del quarto byte è rimasto posto a 1, questa volta dovremo eseguire -(256*61+205) = -15821, cioè dovremo preporre al conteggio il segno di sottrazione.

Visto che ci siamo ormai resi conto del significato dei valori contenuti nei byte, mettiamo all'opera il computer. Tenendo presente che nella linea 1040 avevamo preparato il parametro R che stabilisce il segno positivo o negativo da dare al risultato finale, aggiungiamo le linee che dispongono la conversione dei bit dell'intero e dei bit dei decimali rispettivamente nella parte intera e nella parte decimale del numero da ricostruire. Le due parti verranno da ultimo associate ed R stabilirà il segno (**listato 6**).

Benché si sia parlato anche di parte decimale, abbiamo inserito negli esempi sinora proposti solo numeri interi. Occorre quindi ancora qualche chiarimento, a proposito delle ultime linee del programma, nelle quali il sistema di riconversione dei bit dei decimali è diverso da quello dei bit dell'intero.

Immettiamo in input 0.002929 6875 (attenzione a non sbagliare!). Osserviamo che il computer segnala:

Listato 6.

```
1440 IN=0:IF IN$="0" THEN1480
1450 FOR K=LEN(IN$) TO 1 STEP -1

1460 IF MID$(IN$,K,1)="1"THEN IN=IN+2^(LEN(IN$)-K)
1470 NEXT
1480 DC=0:IF DC$="0"THEN1590
1490 FOR K=1 TO LEN(DC$)
1500 IF MID$(DC$,K,1)="1"THEN DC=DC+(1/2)^K
1510 NEXT
1590 PRINT:PRINT"IL NUMERO INTRODOTTO ERA:"
1610 PRINT(IN+DC)*R
```

Bit dell'intero: 0
Bit dei decimali:0000000011 0000.....

La parte intera, non v'è dubbio, è zero. La parte decimale desta perplessità. Che fanno quei due bit a 1 situati a partire da sinistra al nono e decimo posto?

Per trasformare i bit che rappresentano la parte intera di un numero si sommano, per ogni bit posto a uno, le corrispondenti potenze di 2, crescenti da destra verso sinistra a partire da zero. Invece, per trasformare il contenuto dei bit che rappresentano la parte decimale si sommano, per ogni bit posto a 1, le corrispondenti potenze di 0.5, crescenti da sinistra verso destra a partire da uno. Perciò, nel caso in esame, poiché i bit a 1 occupano il nono e decimo posto iniziando da sinistra, dobbiamo calcolare 0.5^9 + 0.5^10.

Chiediamo al computer in modo diretto di eseguire l'operazione e la risposta sarà, in notazione scientifica, 2.9296875E-03, pari, in notazione standard, al nostro numero.

Il metodo esposto è proprio quello insito nel programma nelle linee da 1490 a1510. Ma perché 0.5? Perché, come nel sistema decimale i numeri (in valore assoluto) minori di uno sono espressi mediante potenze in base 10 con esponente negativo, altrettanto avviene nel sistema binario.

Ma 2 elevato a una potenza negativa è uguale a 0.5 elevato alla stessa potenza positiva. L'uso di 0.5 è opzionale e infatti alla linea 1500, anziché (1/2)^K avremmo potuto scrivere 2^-K.

Non resta che provare ad introdurre nel programma i numeri più disparati per assimilare i meccanismi che governano le variabili.

Constateremo anche che, alle volte, nel ricalcolare il numero, il C64 incorre in approssimazioni dovute ad arrotondamenti nelle sequenze delle operazioni.

Gustavo Fontanella

Amiga qua, Amiga là...

Dopo aver osservato gli aspetti generali delle capacità multitasking di Amiga e come si potessero utilizzare sia da Workbench sia da CLI, ci addentriamo nel sistema operativo di Amiga, per l'esattezza nella libreria Exec, e vediamo come lavorano alcune delle principali funzioni.

Come tutti i calcolatori, anche Amiga dispone di un sistema operativo, ovvero di un'ampia collezione di routine, memorizzate su ROM negli Amiga 500 e 2000, e caricabile da disco nel 1000 (Kickstart).

Questo sistema operativo risulta essere modulare, ovvero le routine che lo compongono sono organizzate in gruppi (moduli) che possono essere librerie o dispositivi. A titolo d'esempio, le procedure contenute nella libreria grafica (Graphics Library) consentono la visualizzazione di testi, elementi grafici e animazioni sullo schermo, mentre il dispositivo audio (Audio Device) permette l'invio, nei canali stereo del calcolatore, di un suono precedentemente campionato o calcolato.

Fra tutte le librerie che sono presenti, a noi interessa in modo particolare la libreria Exec, il cui compito è quello di controllare tutto il funzionamento del multitasking, e quindi di gestire anche le altre librerie. Per questa ragione la libreria Exec è la prima a essere avviata.

Ecco ora l'elenco delle principali mansioni affidate a Exec:

- Gestione della memoria
- Controllo del multitask
- Controllo delle interruzioni
- Invio e ricezione di segnali fra task (e interruzioni)
- Invio e ricezione di messaggi fra task
- Gestione delle librerie
- Gestione dei dispositivi
- Attuazione delle procedure di I/O da parte dei vari dispositivi.

Prima di procedere è opportuno esaminare la gestione delle liste, giacché Exec usa queste strutture in svariati contesti.

Una lista è una successione di elementi, detti nodi, collegati l'uno all'altro in un determinato ordine. Il nodo, allegoricamente, può essere pensato come un doppio gancetto che colleghi un elemento della lista a quello precedente e a quello successivo.

In **tavola 1** è riportata la struttura in C di un generico nodo utilizzato da Exec; essa è reperibile nel file include/exec/nodes.h, disponibile sui diversi sistemi di sviluppo per Amiga. Come si può vedere, ogni singolo nodo contiene, oltre al puntatore al nodo successivo e a quello precedente, anche un indicatore del tipo, priorità di esecuzione e nome dell'elemento collegato a quel nodo.

La lista è a sua volta una struttura che contiene dei puntatori al primo e all'ultimo nodo degli elementi che, collegati in catena, for-

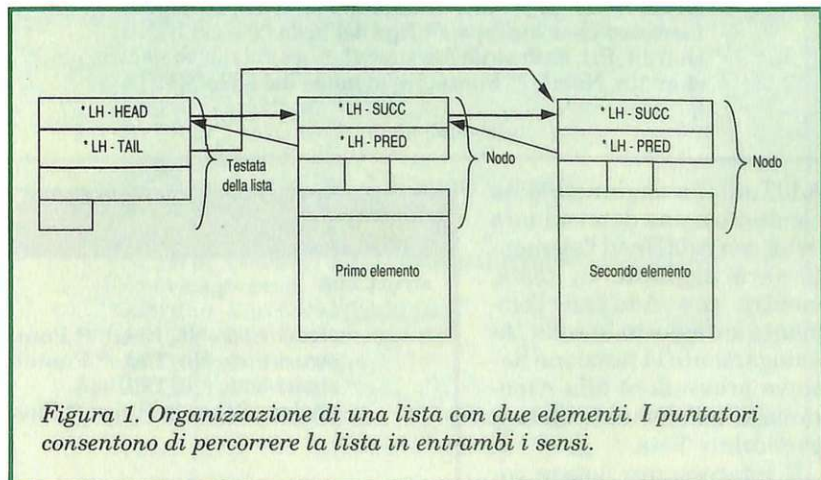



Figura 1. Organizzazione di una lista con due elementi. I puntatori consentono di percorrere la lista in entrambi i sensi.

Figura 2. Collegamento dei blocchi di memoria libera mediante le strutture Mem-Chunk.

mano la lista stessa. La sua rappresentazione in C, contenuta in include/exec/lists.h, è riportata in **tavola 2**. La **figura 1** illustra una generica lista contenente due elementi. Exec, nell'espletare i propri compiti, provvede alla gestione di diverse liste, dette liste di sistema, quali quella preposta al controllo della memoria, al controllo delle diverse task operanti, eccetera. Durante queste operazioni accade spesso che Exec, tramite apposite funzioni, aggiunga o tolga degli elementi da una lista. La stessa operazione può essere eseguita da un programma utente o da un'altra libreria, utilizzando le funzioni AddHead e AddTail che aggiungono un elemento a una determinata lista: con AddHead l'elemento sarà aggiunto in testa, mentre con AddTail l'elemento sarà posto in coda. Analogamente la funzione Remove provvederà alla rimozione di un elemento da una particolare lista.

È interessante notare come tutte queste operazioni di aggiunta e rimozione di elementi da una lista vengano effettuate agendo esclusivamente sui puntatori contenuti nei nodi degli elementi precedente e successivo a quello che si intende aggiungere o rimuovere. Non ci addentriamo ulteriormente nell'uso di queste funzioni, in quanto il nostro scopo è quello di illustrare il funzionamento del multitasking su Amiga e non la programmazione del medesimo in C o in Assembler.

Tavola 1.

```
struct Node
        {
        struct Node *ln_Succ; /* Puntatore al nodo sucessivo */
        struct Node *ln_Pred; /* Puntatore al nodo precedente */
        unsigned char ln_Type; /* Tipo del nodo */
        char ln_Pri; /* Priorità */
        char *ln_Name; /* Puntatore al nome del nodo */
        };
```

Tavola 2.

```
struct List
        {
        struct Node *lh_Head; /* Puntatore primo elemento della lista /*
        struct node *lh_Tail; /* Puntatore ultimo elemento della lista /*
        struct node *lh_TailPred
        unsigned char lh_Type; /* Tipo degli elementi formanti la lista /*
        }
```

Gestione della memoria

La memoria di un sistema multitasking deve essere in grado di ospitare diversi programmi contemporaneamente, con tutte le variabili a essi annesse; inoltre la memoria occupata dai singoli programmi, una volta terminata l'esecuzione, dovrà essere liberata, ovvero resa nuovamente disponibile affinché possa essere nuovamente utilizzata da altri programmi.

Allo scopo la libreria Exec contiene una lista di tutti i blocchi di memoria disponibili (ovvero liberi) e, a mano a mano che i diversi programmi o le diverse routine del sistema operativo chiedono della memoria, questa viene loro dispensata togliendola dalla lista dei blocchi disponibili. Sarà compito degli stessi programmi e delle diverse routine del sistema operativo restituire la memoria a Exec al termine dell'utilizzo, affinché questa torni nuovamente disponibile. Fra le variabili di sistema gestite dalla libreria Exec vi è la MemList, il cui compito è quello di tenere un inventario di tutta la memoria disponibile:

```
struct List MemList;
```

Questa lista ha come elementi delle strutture che rappresentano i diversi tipi di memoria presenti nel calcolatore. Per esempio, in un Amiga 500 non espanso questa lista conterrà un solo elemento, che corrisponderà ai 500 Kb di Chip memory presente. In un Amiga 2000 la lista conterrà due elementi; uno sarà relativo ai primi 500 Kb di Chip memory e un altro rappresenterà i rimanenti 500 Kb di Fast memory.

Le strutture che rappresentano i tipi di memoria presenti, e che

sono quindi elementi della MemList, si chiamano MemHeader (testata della memoria) e contengono tutte le principali informazioni relative a quel tipo di memoria. Eccone la struttura in C contenuta nel file include/exec/memory.h:

struct MemHeader { struct Node mh_Node; /* Nodo di collegamento */ unsigned short mh_Attributes; /* Specifica il tipo di memoria */ struct MemChunk *mh_First; /* Puntatore primo frammento di memoria libero /* APTR mh_Lower; /* Puntatore inizio area di memoria */ APTR mh_Upper; /* Puntatore termine area di memoria */ long mh_Free; /* Quantità totale di memoria libera */ }

La precedente struttura contiene un puntatore al primo frammento di memoria libero (*mh_First); tale frammento a sua volta contiene nei primi 8 byte una struttura così organizzata:

struct MemChunk { struct MemChunk *mc_Next; /* puntatore al prossimo frammento */ unsigned long mc_Bytes; /* quantità di memoria disponibile nel frammento */ }

Dalla struttura si può osservare come le porzioni di memoria libera siano collegate le une alle altre attraverso il puntatore *mc_Next, e come tali blocchi non siano in realtà completamente liberi da ogni informazione in quanto i primi 8 byte saranno sempre occupati dalla struttura MemChunk, che permette a Exec di reperire i diversi blocchi.

Per questa stessa ragione il più piccolo blocco di memoria manipolabile da Exec deve avere un'estensione minima di 8 byte. In realtà tutte le operazioni di allocamento e deallocamento lavorano sempre con estensioni di memoria la cui lunghezza sia un multiplo intero di 8 byte.

La **figura 2** mette in evidenza come sono collegati fra di loro i diversi frammenti di memoria libera, affinché siano reperibili al sistema operativo.

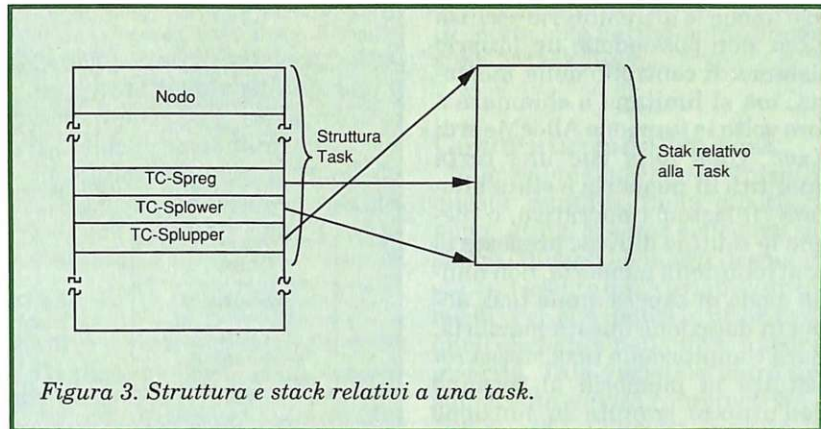



Figura 3. Struttura e stack relativi a una task.

Per allocare e deallocare la memoria, Exec mette a disposizione tutta una serie di funzioni. Le più usate fra queste sono certamente AllocMem e FreeMem, che permettono rispettivamente di allocare e liberare una certa area di memoria. La funzione AllocMem necessita come parametri la quantità e il tipo di memoria che si desidera allocare, e ritorna l'ubicazione della memoria allocata o, qualora l'operazione abbia avuto un esito negativo, ritorna zero (0).

FreeMem necessita invece come parametri l'indirizzo e l'estensione della memoria da restituire.

Altre funzioni di Exec lavorano con la memoria, e permettono l'allocazione e la liberazione di più aree di memoria, anche di tipo diverso, contemporaneamente: AllocEntry e FreeEntry.

Oltre a Exec, altre librerie contengono funzioni per dispensare memoria: per esempio, la libreria grafica contiene la funzione AllocRaster, utile per allocare un BitPlane (piano di bit) per rappresentare immagini sullo schermo. Ovviamente la libreria grafica, e

Tavola 3.

```
struct Process
        {
        struct Task pr_Task; /* Task relativa a questo processo
*/
        struct MsgPort pr_MsgPort;
        short pr_pad;
        BPTR pr_SegList; /* Programma relativo al processo */
        long pr_StakSize; /* dimensioni dello Stak */
        APTR pr_GlobalVector;
        long pr_TaskNum;
        BPTR pr_StakBase;/* inizio dello Stak */
        long pr_Result2;
        BPTR pr_CurrentDir; /* direttory corrente */
        BPTR pr_CIS;
        BPTR pr_COS;
        APTR pr_ConsoleTask; /* dispositivo consolle abbinato
        a questo processo */
        APTR pr_FileSystemTask;
        APTR pr_CLI;
        APTR pr_ReturnAddr;
        APTR pr_PktWait;
        APTR pr_WindowPtr;
        }
```

così anche le altre librerie esclusa Exec, non possiedono un proprio sistema di controllo della memoria, ma si limitano a chiamare a loro volta la funzione AllocMem di Exec. Una volta che una certa quantità di memoria è stata allocata, il sistema operativo, o meglio le routine di Exec preposte al controllo della memoria, non hanno modo di sapere quale task abbia in dotazione questa memoria. Sarà compito della task stessa restituire la memoria al termine dell'utilizzo tramite la funzione FreeMem o FreeEntry.

Strategie di allocamento

Esistono due principali strategie che possono essere seguite da un programma per l'allocamento della memoria. Nel primo modo, detto allocamento statico, tutta la memoria necessaria all'esecuzione di un certo programma viene allocata in un' unica volta all'attivazione del programma stesso e viene liberata, sempre in un'unica operazione, al termine dell'esecuzione dello stesso. Per contro, nell'allocazione dinamica, la memoria viene allocata di volta in volta immediatamente prima dell'uso e liberata al termine dell'utilizzo.

Entrambi i modi presentano vantaggi e svantaggi: l'allocazione statica è certamente più veloce, in quanto svolge tutte le operazioni in una sola volta, tuttavia è necessario sapere a priori la quantità di memoria da allocare. Nell'allocazione dinamica invece, occorrono molte operazioni di allocamento e deallocamento, che comportano un tempo maggiore per l'esecuzione: tuttavia la gestione della memoria è ottimale, in quanto tutta la memoria divenuta inutile viene immediatamente liberata, a beneficio di altri eventuali programmi concorrenti.

Poiché ogni singola task in esecuzione può allocare o deallocare memoria, durante l'esecuzione di più programmi concorrenti (contemporanei) la memoria libera può apparire segmentata, ovvero grandi blocchi di memoria libera possono essere divisi, nella mappa di memoria, da piccoli blocchi di memoria occupata. In questi casi ulteriori tentativi di allocamento di grandi aree di memoria potranno avere un esito fallimentare. Si può porre rimedio a queste situazioni chiedendo la sospensione dell'esecuzione del programma meno importante il quale, deallocando le proprie risorse, libererà parte della memoria.

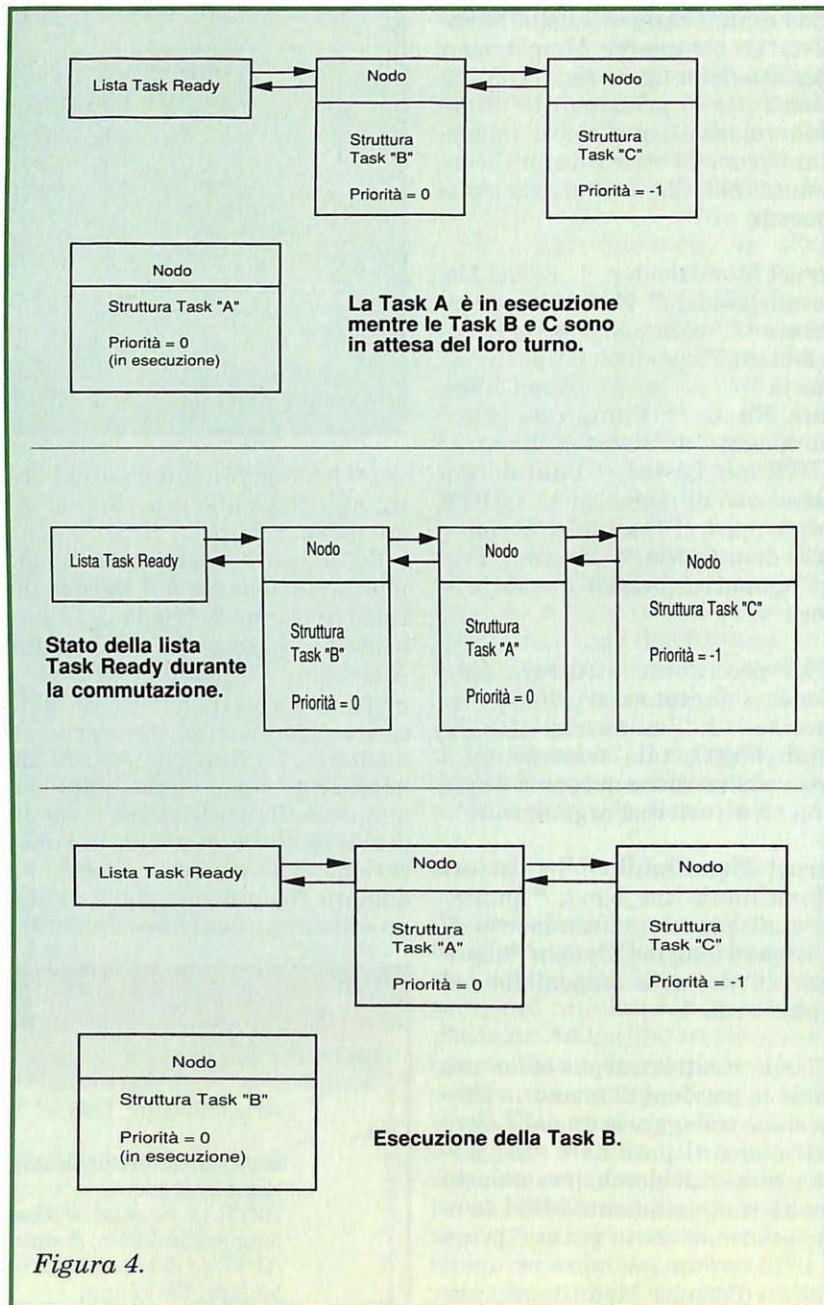



Figura 4.

Processi

Nella scorsa puntata si era visto in che cosa consiste un processo e come Amiga, grazie alle proprie capacità multitasking, è in grado di eseguire più programmi contemporaneamente. Ora esamineremo nei dettagli come questo avviene.

Innanzitutto occorre distinguere fra processo e task, dato che la letteratura riguardante Amiga u-

sa questi termini con significati diversi. Nella scorsa puntata si è visto che cos'è un processo e come è possibile attivarne uno da Workbench o da CLI. Il controllo dei diversi processi che si contendono l'elaborazione del microprocessore viene affidato alla libreriaDOS: infatti, a ogni processo attivo corrisponderà un'apposita struttura detta Process, gestita dalle funzioni della DOS Library.

La struttura menzionata, tratta da include/library/dosextens.h, è riportata in **tavola 3**. A ogni processo è abbinata una task (pr_Task) il cui compito è quello di curarsi dell'esecuzione del programma o dei comandi mandati a quel processo. In termini più semplici, una task, grazie all'omonima struttura, si occupa dell'esecuzione in multitasking di un determinato segmento di codice, mentre un processo si occupa, a un livello molto più alto nella gerarchia del sistema operativo (libreria del DOS), di assicurare l'esecuzione di un certo comando o programma caricandolo e facendolo dialogare con l'utente, anche in maniera interattiva (tramite CLI). Quindi ogni processo dovrà utilizzare una task per poter assolvere i propri compiti, mentre le task, che sono strutture molto più semplici, potranno essere attivate anche al di fuori di qualsiasi processo.

Le task

Concentreremo ora la nostra attenzione sulle singole task e su come queste siano eseguite dal sistema operativo.

Innanzitutto il termine task significa compito, quindi una task è un compito che verrà eseguito dal sistema operativo, magari contemporaneamente ad altri compiti. Ad ogni task, così intesa, corrisponderà nella memoria del calcolatore una struttura task, la cui gestione è affidata, analogamente alle altre funzioni di base, a Exec.

In **tavola 4** è visibile la struttura Task come è contenuta in include/exec/tasks.h. Per poter essere eseguita, ogni task necessita, oltre alla precedente struttura, anche di un segmento di codice che deve guidarne l'esecuzione, e di uno stack nel quale riporre le variabili automatiche e gli indirizzi di ritorno delle varie funzioni.

La **figura 3** evidenzia i collegamenti che vi sono fra la struttura e lo stack relativi a una determinata task.

Generazione di una task

Ovviamente tutte le operazioni per poter generare una task dovranno essere svolte da un'altra task, che chiameremo task genitrice. In contrapposizione la task generata verrà indicata come figlia. Il primo compito al quale dovrà ottemperare la task genitrice è quello di allocare, magari utilizzando le funzioni AllocMem o AllocEntry, tutta la memoria necessaria all'esecuzione della nuova task. Questa memoria comprenderà un area sia per la struttura task, sia per l'allocazione dello stack, dando per scontato che il codice che dovrà eseguire la task figlia sia già presente in memoria.

La seconda operazione consiste nell'inizializzare parzialmente la struttura task; occorrerà infatti inserire in questa struttura i dati relativi alle dimensioni e al posizionamento della memoria precedentemente allocata e destinata allo stack, ovvero compilare le variabili: tc_SPReg, tc_SPLower, tc_SPUpper. Quindi, sempre la task genitrice, deve effettuare una chiamata alla funzione AddTask, che necessita come parametri di un puntatore alla struttura task, di un puntatore al codice che la nuova task dovrà eseguire e, eventualmente, di un puntatore a una routine che, chiamata al momento della rimozione della task , si occuperà di liberare tutte le risorse. La funzione AddTask, della libreria Exec, collegherà la struttura task, utilizzando il nodo presente alla sommità della stessa struttura, alle altre task da eseguire nella lista TaskReady (task pronte). Sarà compito delle procedure automatiche del multitasking, gestite da Exec, mandare in esecuzione nel momento opportuno la task figlia.

Commutazione fra task

Come abbiamo visto all'inizio di questa puntata, nella struttura nodo, utilizzata per collegare insieme i diversi elementi di una lista, è presente una variabile di nome ln_Pri, che indica la priorità di quell'elemento nella lista in cui è inserito. Tale variabile potrà assumere un valore compreso tra -128 e +127. Quando, tramite la funzione AddTask, una task viene inserita alla lista TaskReady, la struttura viene inserita in una posizione dipendente dal valore assunto dalla priorità. Ovvero la TaskReady si presenterà sempre come una lista ordinata, dove le task a maggiore priorità saranno inserite prima di quelle a priorità inferiore. La priorità di una task può comunque essere cambiata in qualsiasi momento utilizzando la funzione SetTaskPri di Exec. In questo caso la struttura task verrà estratta dalla lista TaskReady, verrà modificata la variabile ln_Pri, quindi la struttura verrà reinserita nella lista in una nuova posizione dipendentemente dalla nuova priorità.

Esaminiamo ora, senza addentrarci nei dettagli, come avviene la commutazione fra le diverse task. Per semplificare la cosa immaginiamo che vi siano solo tre task a contendersi il tempo di elaborazione del microprocessore. Supponiamo inoltre che le prime due task, che chiameremo rispettivamente A e B, abbiano priorità 0, mentre la terza task C, abbia una priorità inferiore impostata a -1.

All'inzio della nostra analisi poniamo sia in esecuzione il codice della task A. L'omonima struttura abbinata alla task sarà scollegata dalla lista :TaskReady, che conterrà, collegate tramite gli appositi nodi, esclusivamente le strutture relative alla task B e C.

Questa situazione è rappresentata nella prima parte della **figura 4**. Trascorso il periodo di tempo

Tavola 4.

```
struct task
        {
        struct node tc_Node; /* Nodo di collegamento */
        UBYTE tc_Flags; /* Indicatori vari */
        UBYTE tc_State; /* Stato della task */
        BYTE tc_IDNextCnt; /* Contatore disabilitazione interruzioni */
        BYTE tc_TDNextCnt; /* Contatore disabilitazione multitask */
        ULONG tc_SigAlloc; /* Segnali allocati */
        ULONG tc_SigWait; /* Segnali uscita attesa */
        ULONG tc_SigRecvd; /* Segnali ricevuti */
        ULONG tc_SigExcept; /* Segnali che provocheranno eccezioni */
        UWORD tc_TrapAlloc; /* Trap Allocate */
        UWORD tc_TrapAble;
        APTR tc_ExceptData; /* Puntatore dati procedura d'eccezione */
        APTR tc_ExceptCode; /* Puntatore codice procedura d'eccezione */
        APTR tc_TrapData; /* Puntatore dati procedura di Trap */
        APTR tc_TrapCode; /* Puntatore codice procedura di Trap */
        APTR tc_SPReg; /* Puntatore allo Stak della task */
        APTR tc_SPLower; /* Puntatore limite inferiore dello stak */
        APTR tc_SPUpper; /* Puntatore limite superiore dello stak */
        VOID (*tc_Switch) (); /* Puntatore funzione di commutazione */
        VOID (*tc_Launch) ();
        struct List tc_MemEntry; /* Lista memoria allocata dalla task */
        APTR tc_UserData; /* Puntatore dati utente */
        }
```

(un 16esimo di secondo) prefissato dal sistema operativo per l'esecuzione di ciascuna task, un timer contenuto in una delle due interfacce CIA genererà un segnale d'interruzione (Interrupt) che sospenderà l'esecuzione da parte del microprocessore del codice relativo alla task A, e manderà in esecuzione una funzione di Exec il cui compito sarà quello di predisporre tutto per l'esecuzione della task successiva.

Per l'esattezza, questa funzione del sistema operativo per prima cosa copia il contenuto di tutti i registri del microprocessore nello stack relativo alla stessa task (task A), indi ripone il puntatore allo stack medesimo nella variabile tc_SPReg della struttura Task.

Successivamente la stessa struttura viene inserita, in ordine di priorità, nella lista TaskReady. In questo specifico caso essa finirà al secondo posto, poiché il primo elemento della lista conteneva già una task a priorità 0 (la task B) mentre la task successiva (C) era a priorità -1.

La situazione che si viene a creare è illustrata nella seconda parte della **figura 4**.

Terminata questa operazione, la funzione preleverà, sganciandolo dalla lista, il primo elemento in essa contenuto (la struttura della task B), e leggerà nella variabile tc_SPReg di questa struttura il nuovo valore da inserire nel registro Stack Pointer Utente del microprocessore.

La successiva operazione consisterà nel prelevare dallo stack, utilizzando il registro precedentemente caricato come puntatore, il contenuto di tutti i rimanenti registri, onde fare in modo che essi rispecchino la condizione interna della MPU nel momento dell'interruzione dell'esecuzione della task B a opera della funzione di commutazione.

Successivamente l'elaborazione continuerà con l'esecuzione del codice relativo alla stessa task B, come è evidenziato dalla terza parte della **figura 4**.

Passato un altro 16esimo di secondo, avverrà un'altra interruzione che opererà una nuova commutazione fra le task e manderà in esecuzione la task A, la quale riprenderà dal punto in cui era stata interrotta. Come si può notare, a causa della sua bassa priorità la task C non andrà mai in esecuzione, sino a quando le task A e B non verranno entrambe rimosse dalla lista TaskReady.

Rimozione di una task

Una volta che una task ha terminato il proprio compito, potrà essere rimossa dal sistema affinché non sottragga più tempo all'elaborazione delle altre task.

Questa operazione potrà essere eseguita, tramite la funzione RemTask della libreria Exec, sia dalla stessa task che vuole porre termine alla propria esecuzione, sia da un'altra task, eventualmente la genitrice.

La suddetta funzione del sistema operativo, come parametro necessita dell'indirizzo della struttura task da eliminare. Se tale indirizzo sarà nullo verrà eliminata la stessa task chiamante la funzione.

Se RemTask è utilizzata per eliminare una task diversa dalla chiamante, la funzione si limita a togliere la relativa struttura task dalla lista TaskReady (o dalla lista TaskWait che incontreremo nella prossima puntata).

Viceversa, se la stessa funzione sarà usata da una task che intende autoeliminarsi, la funzione porrà termine all'esecuzione del codice della task stessa senza però riporre la struttura task nella lista TaskReady.

Come si è già visto le singole task sono completamente indipendenti, e l'esecuzione di ognuna viene vista dalle altre come un evento asincrono, ovvero, a meno di esplorare la corrispondente struttura task, sembrerebbe impossibile per una task conoscere a che punto sia arrivata l'elaborazione di un'altra task.

Nella prossima puntata ci occuperemo della sincronizzazione delle diverse task.

Sergio Fiorentini

(continua)

Un occhio per contare

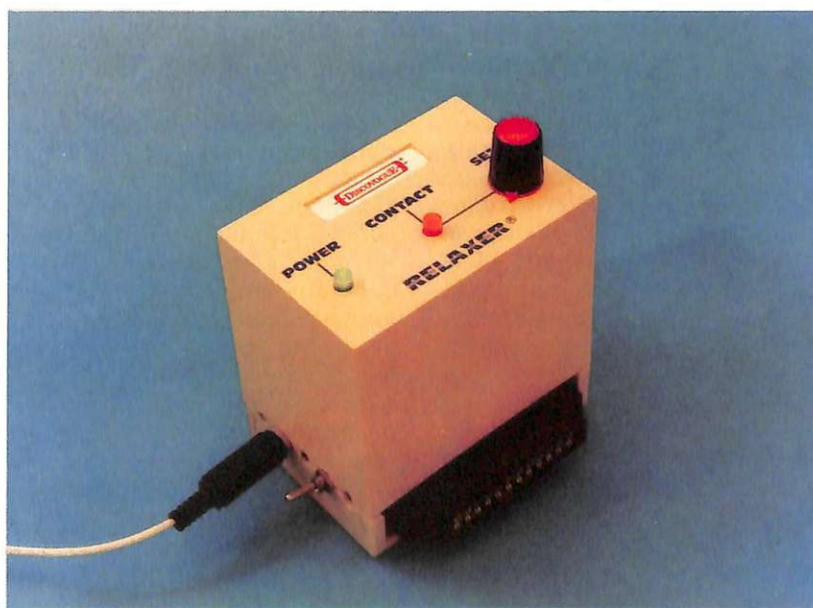
Ora che avete realizzato l'occhio elettronico vi proponiamo come di consueto il software, ovvero il cervello che elabora i dati rilevati dal sensore. L'uso ottimale del sistema si realizza in applicazioni su attrezzature sportive come cyclette e strumenti da palestra, ma anche come impianto di fotofinish per piste d'atletica.

Questo eccezionale programma contiene al suo interno complesse routine di elaborazione e permette di ottimizzare le prestazioni del sistema hardware/software composto dal computer Commodore 64 e dal dispositivo elettronico Palextra (applicato al computer attraverso la porta utente), riuscendo a gestire in modo professionale e totalmente automatico tutte le operazioni di controllo dei segnali di commutazione trasmessi da due distinti sensori ottici ogni volta che qualcuno (o qualcosa) attraversa, anche per brevissimi istanti, i relativi campi di rilevazione compresi tra fotoresistenze dei sensori e appropriate sorgenti luminose di riferimento.

Risultati d'uso ottimali si possono ottenere soprattutto in palestre o spazi-ginnastica di casa, perché speciali contatori digitali e convertitori (impulsi-lunghezze metriche) provvedono a calcolare all'istante, ad esempio, l'ideale tragitto fatto pedalando su una cyclette, oppure la distanza percorsa su un qualsiasi circuito di riferimento (corsa, footing, passi di riscaldamento).

I display forniscono i dati in chilometri, con precisione addirittura al metro, sulla base di parametri di moltiplicazione definibili a piacere dall'utente, da 0 a 99 centimetri (o da 0 a 99 metri). Per il primo input è inizialmente previsto un incremento di 10 centimetri per ogni impulso, con possibilità di conteggio fino a 99.999 passaggi, per un totale gestibile di 9,999 chilometri. L'incremento è tuttavia aumentabile fino a 99 centimetri e in pratica si possono conteggiare distanze fino a 98,999 chilometri: definendo allora un incremento base di 94 centimetri e collegando il relativo sensore alla ruota di qualsiasi cyclette, a 15 centimetri dal centro (formula geometrica circonferenza/raggio), e fissando un pezzo di cartoncino nero (schermo) su un punto mobile equidistante che vada a passare tra luce e fotoresistenza, creando un impulso attivatore a ogni pedalata ciclica, si può ottenere la misurazione digitale e precisa del percorso realizzato (normalmente dopo 10 pedalate saranno conteggiati 0,009 chilometri, l'equivalente di 9,40 metri).

Per il secondo input l'incremento di default previsto è invece di 10 metri per ogni impulso, con possibilità di conteggio fino a 9.999 passaggi, per un totale di 99,99 chilometri. Anche in questo caso la scelta parametrica è va-

riabile (fino a 99 metri) in quanto il canale di input è pensato per abbinamenti a percorsi di qualsiasi genere e lunghezza, ricavabili al limite anche nella stanza di un appartamento.

Si pensi a un circuito quadrangolare di 2,5 + 2 + 2,5 + 2 metri (perimetro 9 metri): è sufficiente in questo caso fissare il fattore di moltiplicazione a 9, cioè il totale percorso per ogni giro. Il massimo gestibile è un iter di ben 99 metri, per un totale di 989,901 chilometri.

Posizionando opportunamente il sensore di rilevazione in un punto qualsiasi del percorso, che diventa così la linea di partenza, di conteggio intermedio e di arrivo, si possono effettuare corse, allenamenti di marcia, giri di riscaldamento.

In pratica il software è in grado di elaborare tutti i segnali che determinano, attraverso i due input PB0 e PB1 della porta utente del computer, la configurazione elettronica di riferimento (attivazione sul primo canale, sul secondo o anche su entrambi contemporaneamente).

Funzionamento e uso

Terminata l'operazione di caricamento compare la videata-monitor a fondo e bordo di colore nero, che rimane invariata nella sua parte grafica e strutturale durante tutte le fasi di elaborazione delle routines.

Nella parte superiore dello schermo sono presenti il nome Palextra 2 (in vari colori) e il dataset di copyright (in marrone). L'area video rimanente viene suddivisa in due distinti settori, graficamente simili e separati da una linea verticale gialla: sulla sinistra, in verde chiaro, c'è quello relativo al primo canale (Input 1, linea PB1, cyclette e simili), mentre l'altro è sulla destra, in rosso chiaro, e monitorizza il secondo canale (Input 2, linea PB0, percorsi e circuiti).

Relativamente al primo input il parametro base di moltiplicazione è definito in centimetri: è variabile a piacere da un minimo di 0 a un massimo di 99 ed è inizialmente autodefinito a 10. Il riquadro della distanza percorsa (in chilometri) contiene un valore nullo (0,000), proprio perché è azzerato il numero dei passaggi rilevati.

Il menù riportato a fondo video ricorda all'utente che per dare inizio al conteggio è sufficiente premere il tasto A: a ogni impulso trasmesso dall'apparecchio Palextra il contatore Passaggi incrementa di 1 (fino a un massimo di 99.999), mentre la distanza accreditata è il risultato della moltiplicazione tra il valore Passaggi e il parametro base (10 centimetri inizialmente), trasformato poi in chilometri. Con il tasto Z si può fermare il conteggio in qualsiasi momento e i dati rimangono visualizzati: riprendendo un nuovo conteggio (A) si causerà anche un reset automatico totale. Con il tasto Q si può variare (anche mentre è in corso un monitoraggio) il valore del fattore di moltiplicazione, semplicemente inserendo quello desiderato (da 0 a 99), usando a tal scopo i tasti numerici: la cifra viene formata sequenzialmente, con scrolling da destra verso sinistra, quindi premendo 4 con un 25 sul video si ottiene poi 54 (il 2 scrolla e scompare dal display). I valori minori di 10 potranno essere inseriti facendo precedere lo zero (0).

Quando si opta per una variazione parametrica (tasto Q) si causa sempre uno stop di conteggio seguito dal reset totale.

Per il secondo input il parametro base di moltiplicazione è invece definito in metri (valore di default 10): anche in questo caso la proiezione iniziale è quella di azzeramento completo. Vale, per questo secondo canale, il modo di procedere già visto per il primo, con la differenza che cambiano ovviamente i tre tasti abilitati, come mostra il relativo menù a fondo video.

Disponendo con il tasto S l'inizio, il contatore Passaggi incrementa di 1 (fino a un massimo di 9.999) con ogni impulso di input, mentre la distanza accreditata è il risultato della moltiplicazione tra il valore Passaggi e l'incremento scelto, trasformato poi in chilometri. Il tasto X permette di fermare il conteggio e di lasciare visualizzati tutti i dati: reset totale e nuovo conteggio si azionano ripremendo S. Il valore del fattore di moltiplicazione si modifica selezionando l'apposita routine col tasto W e inserendo le cifre con lo stesso criterio già visto per il primo input. Si può variare da un minimo di 0 a un massimo di 99 metri.

Per un buon funzionamento di programma e apparecchio Palextra è necessario posizionare al meglio i sensori di rilevazione, in modo che le lucette di riferimento colpiscano direttamente le relative fotoresistenze, e che ogni impulso di commutazione inviato agli input sia la conseguenza del passaggio, anche velocissimo, di uno schermo (persona in transito o filtro in movimento) nel campo di rilevazione stesso.

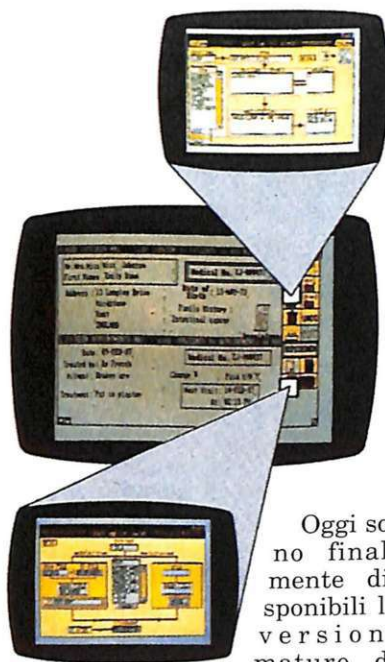
Il primo canale (Input 1) va riservato ad abbinamenti per conteggi frequenti e di breve distanza (centimetri), come cyclette (riferendosi al perno di rotazione della ruota), mentre il secondo canale (Input 2) è adatto a controlli su percorsi aventi uno sviluppo perimetrale di 99 metri come massimo.

Durante il funzionamento del programma anche il dispositivo Palextra svolge un'appropriata azione di monitoraggio assai utile all'utente, tramite le segnalazioni dei tre led di colore verde (Power) e giallo (Input 1 e Input 2) che segnalano, rispettivamente, il corretto funzionamento dell'apparecchio e le commutazioni di livello logico trasmesse dai due sensori periferici ogni volta che uno schermo attraversa, anche per pochi attimi, il campo di rilevazione compreso tra fotoresistenza e sorgente luminosa di riferimento.

Daniele Malavasi

È opinione abbastanza diffusa che l'Amiga sia un computer poco serio, poco adatto all'uso professionale, perché l'enorme quantità di software di cui dispone è, in maggioranza, d'intrattenimento. Quello che appare come uno stupendo giocattolo però, si rivela, all'occhio di chi si pone al di fuori dei luoghi comuni, un potente e disinvolto strumento professionale.

Amiga è anche professionale

Per molto tempo si è lamentata l'assenza di software applicativo a livello veramente professionale per Amiga. Sul versante dei fogli di calcolo questo vuoto è stato colmato dal Vip professional e dal famoso Maxiplan. Per i database si sono visti degli eccellenti prodotti come Microfiche Filer e i database della Softwood, tutti però indirizzati a un uso domestico, senza la pretesa di prestazioni dai grandi numeri. La prima versione di Aquisition, il database della Taurus, e la versione personal del Superbase hanno indicato che qualcosa di veramente buono era prossimo alla realizzazione.

Oggi sono finalmente disponibili le versioni mature di entrambi i prodotti, cui vogliamo dare un'occhiata approfondita.

Superbase Professional

Il programma ricalca fedelmente l'impostazione del Superbase personal, programma che molti di voi avranno sicuramente provato e apprezzato, essendo stato tradotto in italiano e distribuito, insieme a svariati altri programmi, nello "scrigno del software" legato alla distribuzione in Italia del modello 500 di Amiga.

Superbase si distingueva dalla concorrenza per la sua capacità di lavorare su file da disco, consentendo di creare archivi consistenti senza dover impegnare un capitale in espansioni di memoria. Tutto questo togliendo poco o nulla alla velocità e alla praticità di database operanti in memoria come Mi Amigafile o Microfiche filer. In aggiunta Superbase offriva una più flessibile e potente capacità di gestione dei dati archiviati.

Con la distribuzione della release 2, il tanto atteso Superbase professional, si sono aggiunti un linguaggio di programmazione e un form editor. Il linguaggio, chiamato DML (Database Management Language) è simile al Basic e viene interpretato dallo stesso programma principale di Superbase.

Alle strutture di comando tipiche dei Basic avanzati (if-then-else, call eccetera), il DML aggiunge tutti i comandi necessari a una duttile e completa gestione dei file generati da Superbase, nonché a una interazione con l'utente attraverso i menù a discesa e i requesters di intuition.

I programmi in DML possono essere direttamente scritti dall'interno di Superbase con il text editor incorporato, oppure letti da file ASCII generati esternamente.

Il form editor è un programma indipendente che può, memoria permettendo, essere richiamato dall'interno di Superbase come se fosse un editor interno al modulo principale del programma. Esso consente di disegnare delle maschere attraverso cui presentare i dati all'utente.

Sullo schermo possono coesistere diverse finestre grafiche, dati da campi interni appartenenti a file differenti e finestre con testi da file ASCII esterni, il tutto condito da intestazioni e decorazioni a testo e in grafica nei formati standard di Amiga.

Ogni immagine IFF, o set di caratteri, è utilizzabile a questo scopo. Un form non è però solamente una maschera, ma uno strumento di interfaccia verso l'utente che consente di filtrare e manipolare i dati visualizzati o immessi in maniera intelligente e completamente invisibile all'utente.

Un form ben progettato, usato in sintonia con un sottostante programma scritto in DML, consente di fornire all'utente un sistema di gestione dati estremamente gradevole e potente, senza che questi si accorga della presenza di Superbase al di sotto della applicazione.

A queste due innovazioni si aggiungono una serie di migliorie alle funzioni già esistenti, che portano tutto il pacchetto a uno standard professionale di elevata qualità.

Al di là delle funzioni di database, Superbase si caratterizza per l'estrema semplicità e compattezza. Tutte le funzioni sono accessibili attraverso il modulo principale del programma e appaiono accessibili in maniera molto funzionale.

La veste grafica del programma è sobria ma efficace: Superbase fa uso di tutte le possibilità di Intuition per rendere chiara e agevole l'interazione con l'utente, senza però alcuna esagerazione.

Il programma può funzionare anche con il solo disk drive interno di Amiga, ma due drive renderanno le operazioni molto più agevoli.

Su una macchina con solo 512 Kb di RAM è molto difficile riuscire a lavorare con forms, campi grafici e sonori, senza incorrere in seri problemi, compresa qualche "Guru Meditation" (blocco totale del sistema) alla prima mossa particolarmente azzardata.

Se avete una macchina espansa ad almeno 1 Mb, Superbase è senza alcun dubbio un programma splendido. Viene distribuito su tre dischi ed è protetto con l'inattaccabile chiave hardware (dongle). C'è da aspettarsi che anch'esso, come è stato fatto per Superbase personal, venga al più presto tradotto in italiano e distribuito attraverso la Commodore italiana.

Aquisition

Sulla carta si tratta di un applicativo letteralmente esagerato, di gran lunga più potente di tutti i pacchetti di gestione dati che abbiamo visto finora.

Aquisition ha le stesse possibilità di Superbase, ma tutte ampliate e rese più duttili da una miriade di opzioni.

Anche Aquisition ha un linguaggio di programmazione, noto come Acom, e naturalmente si tratta di una specie di Basic dotato dei comandi per accedere alle funzioni di Aquisition. Acom è

tanto potente che tra gli esempi acclusi al pacchetto ci sono un gioco (un piccolo Trivial Pursuit) ed esempi di animazioni grafiche. L'intero pacchetto è realizzato con una cura maniacale della grafica, che si esprime in icone animate, requesters dipinti, pergamene medioevali e un pointer animato ritraente un tipico poliziotto londinese al posto della classica nuvoletta "zzzz".

Per realizzare tutte queste possibilità il programma è in realtà spezzato in una serie di moduli separati. Se vogliamo per esempio creare un archivio dati relazionale, è necessario passare per il modulo Creating per dare ad Aquisition le definizioni dei file di cui sarà composto il database e procedere con il modulo Bridging, all'interno del quale vengono definite tutte le relazioni che collegano i diversi file del nostro archivio.

Per rendere più gradevole l'uso dell'archivio si può utilizzare il modulo Pasting, che consente di creare schermate e maschere grafiche attraverso le quali presentare i dati estratti dai file in maniera analoga a quanto è possibile fare attraverso il form editor di Superbase.

Per usare l'archivio dati così preparato sono disponibili i moduli Filing e Reporting, rispettivamente dedicati all'introduzione di dati nei file e alla ricerca ed estrazione di informazioni.

Completa il programma una serie di moduli di servizio mediante i quali effettuare tutte le procedure di installazione del programma e di manutenzione della base dati, nonché una piccola calcolatrice che sorprendentemente è in grado di lavorare direttamente sui campi dati del record corrente e di interpretare linee di programma Acom.

Naturalmente i risultati ottenuti e le righe di programma scritte possono essere direttamente inseriti nel database.

Aquisition 1.3 viene distribuito su tre floppy e non è in alcun modo protetto dalla copia. Grazie a questa divisione in moduli è perfettamente possibile usare Aquisition su una macchina con solo 512 Kb di memoria senza alcun problema, ma questa organizzazione dispersa va a scapito della semplicità d'uso del pacchetto.

Funzioni che in Superbase si attivano con un tasto, in Aquisition richiedono spesso di cambiare modulo dopo essere tornati al workbench. Alcune funzioni sono presenti in più di un modulo, ma a un primo impatto questo rende l'insieme, se possibile, ancora più caotico. È necessaria una certa pratica per imparare che cosa si può fare e da quale modulo si può farlo.

Inoltre è opportuno avere ben presente il modello di database che si vuole implementare prima di mettersi al lavoro, altrimenti si finisce inevitabilmente con il disperdersi tra le mille possibilità di questo software. In altre parole, se necessitate di Aquisition per implementare una applicazione complessa, dovete anche avere una certa conoscenza della teoria che sta dietro i database relazionali.

D'altra parte è implicito che Aquisition non è destinato a un'utenza hobbistica, e l'aspetto visivo frivolo del programma non deve fare minimamente sottovalutare la sua serietà e la sua potenza.

Oltre a questa sensazione di sentirsi un po' perso tra le mille facce di Aquisition, possiamo solo aggiungere che, nel corso delle prove effettuate, abbiamo notato che il programma ha la tendenza a "mettersi a meditare" senza avvisare l'utente.

Ogni gesto di impazienza dello sprovveduto utente porta invariabilmente sulla strada di un inchiodamento del programma o di una "Guru Meditation" del sistema operativo.

In conclusione, Aquisition è di una potenza esagerata rispetto alle necessità di un normale utente di personal computer. Il programma è dimensionato per le necessità di una piccola ditta o di un grosso ufficio, ma chi ha masse di dati a misura di Aquisition avrebbe sicuramente bisogno, se non ne è già dotato, di un sistema informatico un po' più grosso di un Amiga, su cui far girare anche il software paghe e contributi, fatturazione, magazzino, il tutto possibilmente integrato. Inoltre l'estrema ricercatezza visiva del programma va a discapito della funzionalità e della semplicità di uso di Aquisition.

Facciamo un esempio concreto: l'utente che deve caricare un file da manipolare ha bisogno di un file requester funzionale e veloce, e può fare tranquillamente a meno di una splendida ma poco leggibile pergamena scritta e decorata in gotico flamboyant, come quelle che sono di regola in Aquisition.

Conclusioni

La conclusione di questo confronto è che entrambi i programmi sono splendidi e sufficientemente potenti per soddisfare ogni necessità.

Superbase vince il confronto per praticità d'uso ed efficienza, mentre Aquisition ha un surplus di potenza e di veste esteriore impressionanti, ma di utilità discutibile, almeno fintanto che ci saranno in giro pochi Amiga con hard disk da 40 Mb.

Entrambi i programmi sono a uno stadio di maturità che li rende perfettamente utilizzabili, anche se qualche ruvidità da correggere la si può trovare nell'uno come nell'altro.

Il rapporto prezzo-prestazioni favorisce Superbase, ma la scelta dipende dalla complessità del modello di applicazione che volete generare.

Ciò che veramente manca a entrambi i programmi è la presenza di un compilatore che trasformi i programmi in DM1 o Acom in applicazioni indipendenti, caratteristica che ha fatto la fortuna dell'accoppiata Ms-Dos/dBase III.

Per il momento nessuno dei due contendenti ne parla, ma la speranza, si sa, è l'ultima a morire.

Andrea Zanna

Linee dati

Ho visto spesso dei programmi Basic in cui sono contenute linee introdotte da un'istruzone DATA. Sul manuale allegato al computer ho letto che queste linee di programma specificano una serie di dati e sono lette mediante l'istruzione READ. Alcune volte, però, vi sono più istruzioni READ e mi chiedo come il computer possa sapere a quale linea di dati esse si riferiscono.

Adriano Signorini
Varese

Il sistema con cui il computer legge le sequenze di dati inseriti nelle linee DATA è molto più semplice di quanto potrebbe sembrare. La prima istruzione READ del programma, dopo il RUN legge il primo dato della prima linea dati. Ogni volta che il programma esegue un'istruzione READ legge il dato successivo a quello letto nell'esecuzione della stessa istruzione precedente. Il fatto che induce in errore è che i dati possono essere suddivisi in linee di programma tutte identificate dall'istruzione DATA, ma il computer li considera tutti in sequenza continua. Per ricordarsi quale dato ha letto per ultimo (per esempio il terzo della seconda linea DATA), il computer aggiorna un puntatore, ovvero delle celle di memoria che contengono l'indirizzo dell'ultimo dato letto. Vediamo un esempio:

```
10 FOR N=1 TO 5:READ A:NEXT
20 FOR N=1 TO 4:READ A:NEXT
30 DATA 1,2,3,4
40 DATA 5,6,7,8,9
```

La linea 10 contiene la prima istruzione READ che legge il primo dato della prima linea dati (30) ponendolo nella variabile numerica A. Il primo loop, in linea 10, esegue cinque volte l'istruzione READ. Al termine del loop i dati dall'1 al 5 saranno stati letti uno di seguito all'altro, anche se sono posti su due linee di programma diferenti. La linea 20, analoga alla 10, esegue altre quattro volte l'istruzione READ leggendo i dati da 6 a 9. Prima di eseguire l'istruzione la prima volta, il programma ha consultato il puntatore sopra accennato, ha saputo che l'ultimo dato letto da un'istruzione READ era il primo della linea 40, e ha letto così il successivo.

Se avete capito il discorso fatto, vi chiederete sicuramente che cosa accadrebbe se, per esempio, si chiedesse al computer di eseguire 9 volte l'istruzione READ quando i dati sono solo 8. Contrariamente a quanto si potrebbe supporre, la lettura dei dati non riprende dal primo della serie, ma si interrompe con il messaggio: "out of data".

Ogni volta che si esegue il Run del programma Basic il puntatore ai dati viene azzerato, cioè indirizzato al primo della serie. In alcuni casi può essere necessario azzerare il puntatore senza arrestare il programma, per dare nuovamente il Run. In questo caso si ricorre all'istruzione Restoreche, quando eseguita, riporta l'attenzione del computer sulla prima istruzione della prima linea dati del programma.

Per una RadioElettronica & Computer migliore

Aiutaci a rendere Radio Elettronica & Computer sempre più interessante. Compila questo breve questionario, ritaglialo e invialo a: Radio Elettronica & Computer - Questionario - Via Ferri 6 - 20092 Cinisello Balsamo (MI).

1. *Quanti anni hai?*
① 12 - 17
② 18 - 24
③ 25 - 34
④ 35 - 49
⑤ Oltre 50

2. *Usi il computer soprattutto per?*
① Lavoro
② Studio
③ Programmazione

3. *Con quale regolarità segui questa rivista?*
① Tutti i numeri
② Il 50% dei numeri
③ Quasi nessuno

4. *Quali argomenti preferisci fra quelli trattati su RE&C?*
① Consigli di programmazione avanzata
② Utility e loro utilizzo
③ Videogames e intrattenimento
④ Prove hardware
⑤ Recensioni varie
⑥ Angolo di Amiga
⑦ Fai da te

5. *Quali argomenti vorresti fossero trattati*
① Articoli per uso didattico
② Corsi di programmazione
③ Articoli per principianti
④ Articoli per C128, C16, Plus 4

6. *Quali altri computer usi oltre al C64?*
① Commodore Amiga
② Apple II
③ Apple Macintosh
④ Ms-Dos
⑤ Altri

7. *Quali periferiche completano il tuo computer?*
① Drive
② Stampante
③ Monitor
④ Modem
⑤ Altro

10. *Quanti programmi acquisti ogni anno?*
① 0-10
② 11-20
③ Più di 20

11. *Che tipo di programmi usi di più?*
① Giochi
② Utility
③ Didattici

12. *Possiedi un Commodore Amiga?*
① Sì ② No

13. *Hai intenzione di acquistarlo?*
① Sì ② No

14. *A scuola usi il computer?*
① Sì ② No

15. *Se Sì, è un C 64?*
① Sì ② No

14. *Qual è la rivista di informatica che leggi di più?*
① Radio Elettronica&Computer
② Commodore Computer Club
③ Commodore Professional
④ Altre (quali...)

DATI PERSONALI

Nome ....

Cognome ....

Indirizzo..

Cap .... Città .... Prov. ....

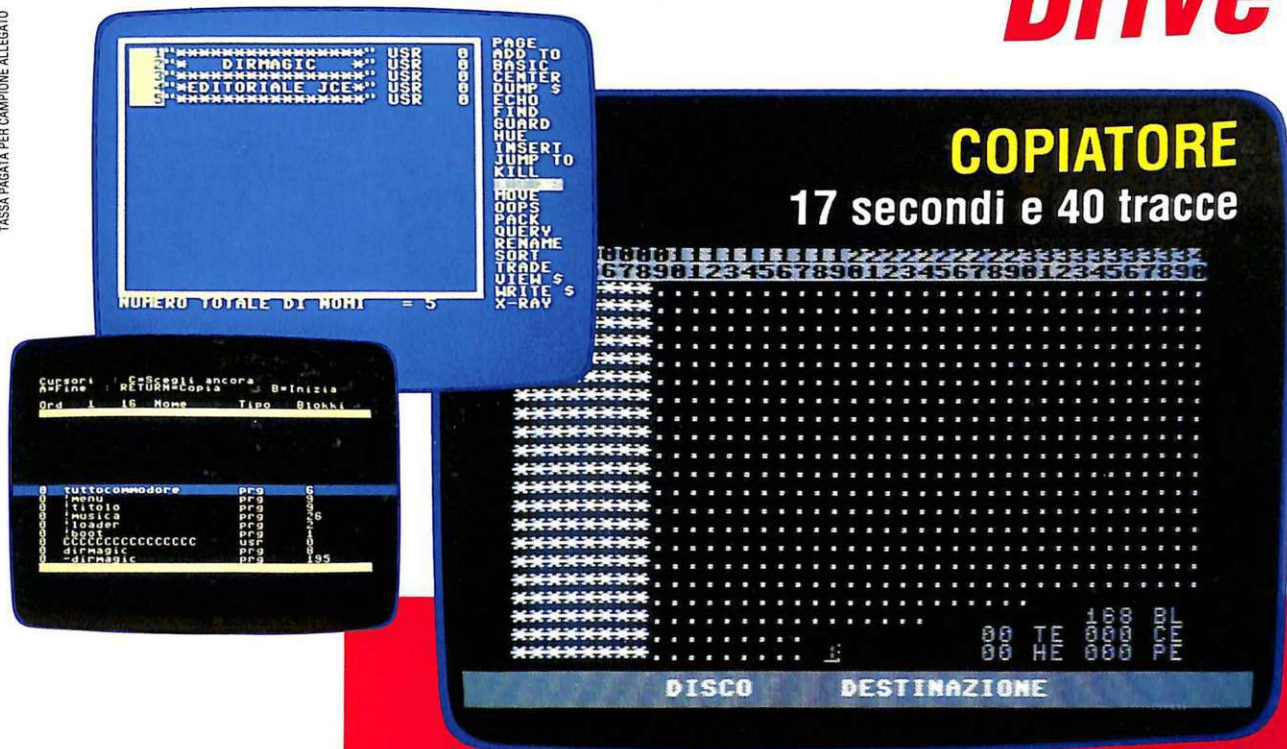
Sped. in Abb. Postale Gr. III/70%

Tutto COMMODORE Drive

Anno III - Numero 23 - Maggio 1989 - L.13.000

TASSA PAGATA PER CAMPIONE ALLEGATO

COPIATORE
17 secondi e 40 tracce

DISCO DESTINAZIONE

FORMATTAZIONE
Tracce speciali

DUPLICAZIONE
Flash di 225 blocchi

DIRECTORY
Esteticamente accessoriata

TRACCIA 41
L'ultima conquista